

Systementwicklungsprojekt

**Construction of a mobile navigation wheel using
Bluetooth**

Dominic Tietz

Aufgabensteller: Univ-Prof. Gudrun Klinker, Ph.D.

Betreuer: Dipl.-Inf. Björn Schwerdtfeger

Abgabedatum: 14. Februar 2007

Contents

1. Introduction	3
1.1. Motivation	3
1.2. Objectives	3
2. Related Work	5
2.1. Hardware	5
2.2. Software	6
3. Wheel Construction	7
3.1. The Ring Housing	7
3.2. The Electronics Housing	9
3.3. The final finger mounted Wheel	11
4. Menu Visualisation	12
4.1. Menu Structure	12
4.2. Menu visualization demos	13
4.2.1. The "RollMenu"	16
4.2.2. The "ClassicMenu"	17
4.2.3. The "SlideMenu"	18
4.2.4. The "EllipseMenu"	18
5. Evaluation	20
6. Conclusion	28

1. Introduction

All computer users have to navigate in different kinds of menus or have to navigate in documents like in an office application. But this times more and more new computer interface technology is arising like for augmented reality applications for example. With these technologies the user is no longer fixed in front of his computer, but can move free in space and interact directly with the virtual environment.

This Systementwicklungsprojekt deals with exactly that topic. So, a mobile navigation wheel had to be designed, which is wirelessly connected to the computer. The next step was to develop a bunch of menu visualizations which can be controlled by the navigation wheel. As a final step a usability test was developed, to evaluate the mentioned concepts.

1.1 Motivation

In order to make menu navigation in such systems like the above mentioned more easy and more comfortable, this Systementwicklungsprojekt tries to discover a new mobile input Device for menu navigation. Because almost every computer has a mouse with a wheel, most people should be familiar with that concept. So, to navigate in all kinds of menu visualizations, a wheel mounted on the finger seems to be quite appropriate for such tasks.

1.2 Objectives

The project is separated in two parts. The first is the hardware side, the second the software side.

On hardware side, the goals of this project are to create a clickable mouse wheel mounted on the finger. The wheel has to be wirelessly connected to the computer. In order to construct a

finger mounted device one has to think about how to mount it and what is the best method to navigate with the wheel. How these tasks are done is discussed in detail later.

On software side the goals are to implement a visualization demo which can be controlled by the navigation wheel. To find the best and most intuitive menu visualization and combination of clicking and scrolling, four different visualization demos have to be implemented and tested by several users. All visualizations have to be configurable in concerns to direction of scrolling and combinations of clicks and scrolls.

In order to get good results the program has to log all user activity in background to be able to evaluate the users behavior and get the best solution.

2. Related Work

In this part I will take a look at related work in means of finger or hand mounted user interfaces and menu visualizations. A lot of the things mentioned in this chapter inspired me in implementing specific concepts and solutions.

2.1 Hardware

There are two devices I want to mention here: The “Ring Mouse” (Figure 1) and the “Finger Sleeve” (Figure 2). Both of them are finger mounted devices. The main difference between those and my wheel device is, that those devices just have buttons, but no possibility to navigate like it is possible with a wheel. Instead, the “Ring Mouse” has an acoustic tracking system with which can be navigated freely in space.

The “Finger Sleeve” also has buttons and a tracking system. This tracking system is not acoustic but electromagnetic.



Figure 1: Ring Mouse

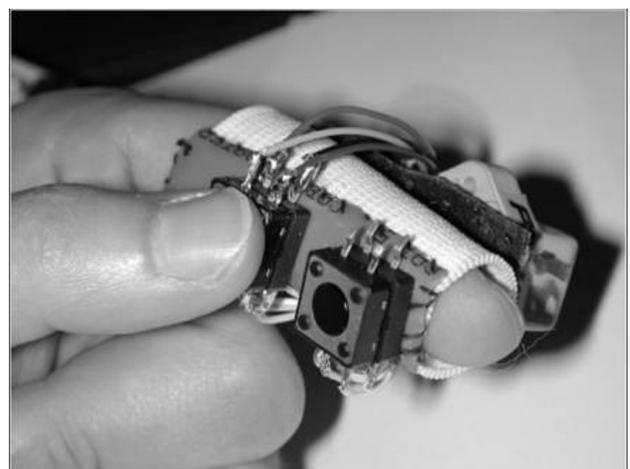


Figure 2: Finger Sleeve

2.2 Software

A lot of 1, 2 or 3 dimensional menu structures are already in use. A lot of them are very well known by most people. For example the menu structure every windows-based system has to offer.

Another menu visualization might also be very familiar: An elliptic shaped menu visualization. Each menu item is placed on a circle, the caption of each item facing the user. Viewed from a specific angle the circle looks like an ellipse. It is a very popular menu visualization in DVD-Players or multimedia systems like the Xbox.

A mixture of a Hardware and Software visualization is the TUIster Project (Figure 3), developed at the Ludwig Maximilians University in Munich. As one can see, some small displays are mounted on a cylinder. By rotating the cylinder one can select the different menu items represented by the different displays. With the help of Tuister one is able to represent a hiracial menu structure.



Figure 3: TUIster Project

3. Wheel Construction

The first step in constructing the wheel and its electronics was to find an existing input device to modify. As one can imagine, this was an easy task: I just took a bluetooth mouse. It has all we need: A wheel to scroll up and down and is connected to the computer using a wireless connection. The only two conditions the mouse had to fulfill was a specific construction scheme of the rotation sensor mounted to the wheel, so that it can be easily put into the ring housing and it had to be as small as possible, so that the actual ring doesn't get too big. On Figure 4 you can see the mouse I chose.



Figure 4: Anycom Bluetooth Mouse

3.1 The Ring Housing

The biggest challenge in creating the ring housing was to keep its design very small. That made it quite difficult to construct it because the milling and drilling had to be very precise. As the material, I chose plates made of PVC. This material is easier to handle as aluminium or stainless steel, but also very solid and lighter.

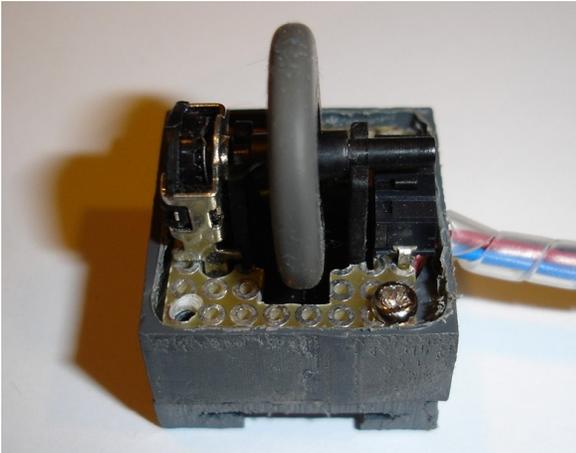


Figure 5: Conductor plate in housing

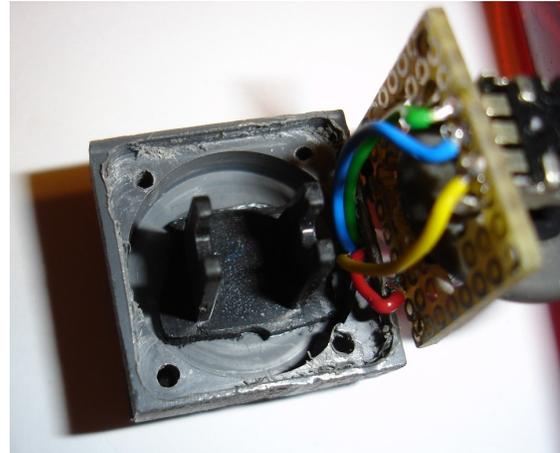


Figure 6: Opened ring housing

To find out the dimensions for the PVC plate, I had to solder out the wheel and the sensors and put them on a new conductor board. You can see the new conductor board with the wheel and sensors on it in Figure 5.

The dimension of the PVC plate depends a lot from the dimensions of the mouse wheel, the rotation sensor and the holding of the wheel. This brought me to the minimum size of 20 x 20 x 10 mm. After sawing a piece of the right dimension out of the PVC plate, I milled a round hole in the middle of that piece, so that the wheel will fit in the hole. Because the plate is quadratic and the hole is round, there is enough space at the corners to drill holes for screws to fix the wheel and the conductor board on it. After that, I lowered the corners of the plate as much as possible to get the wheel as deep as possible into the housing. To give the wheel hold, I sawed out the holding for the wheel out of the mouse housing and glued it in the middle of the hole in the PVC plate. This is shown on Figure 6. After connecting the sensors to small wires and leading them out of the housing, the conductor plate can be mounted in the housing with some screws. How it is mounted is also shown in Figure 5 and 6.



Figure 7: The holding for the finger

Now its time to think about how this construction is mounted on the finger. The first idea that came to my mind was to use an elastic tape. But this is too flexible, so that when clicking and rotating the wheel it always moves on the finger, which makes it quite uncomfortable. So I decided to construct a holding like in Figure 7. As one can see, it consists of a ring made of PVC and a guide rail to mount the ring on it. This makes the ring very flexible, because it can be worn in two ways, each rotated by 90 degrees.

3.2 The Electronics Housing

Construction of the electronics housing wasn't as challenging as the ring housing, because the housing is much bigger, so it is easier to handle. I used the same material as before with the ring.

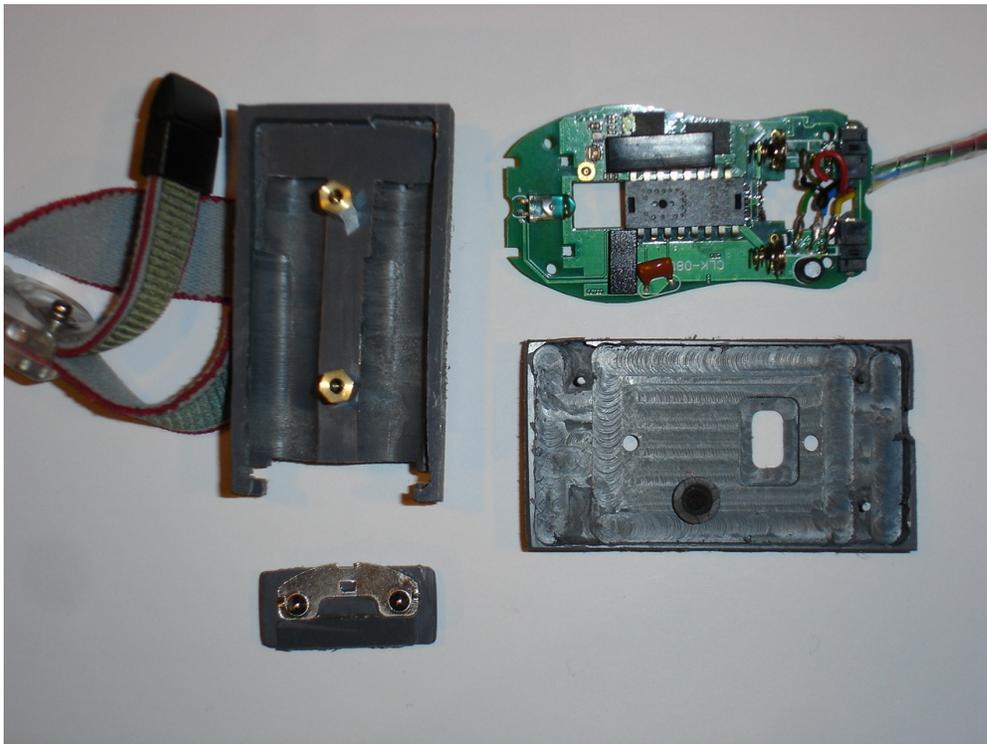


Figure 8: The electronics and its housing

The first step was to create the bottom plate. So I took the conductor plate of the mouse and marked the mounting points. Then I milled the right shape into the PVC plate, as shown in Figure 8 on the lower right side. After some refinements the conductor plate matched the mounting points and the cover could be created.

The cover is a quite simple construction, as you can see in Figure 8. It is just a PVC block with two holes for the batteries. The cover is fixed on the base plate with two spacers many people know from their mainboards to mount them in the case. At the back of the cover is a little shutter to be able to change the batteries. The shutter is also shown on the lower left side of Figure 8.

To mount the housing on the arm I used a anti-static band and screwed it on the back of the cover. Quite simple, but works best.

Of course, the electronics is then connected to the wires mentioned in the description of the wheel housing.

3.3 The final finger mounted Wheel

Finally Figure 9 shows the finished input device mounted at an arm. As you can see, I wrapped the wires with a spiral tube to make it more compact.

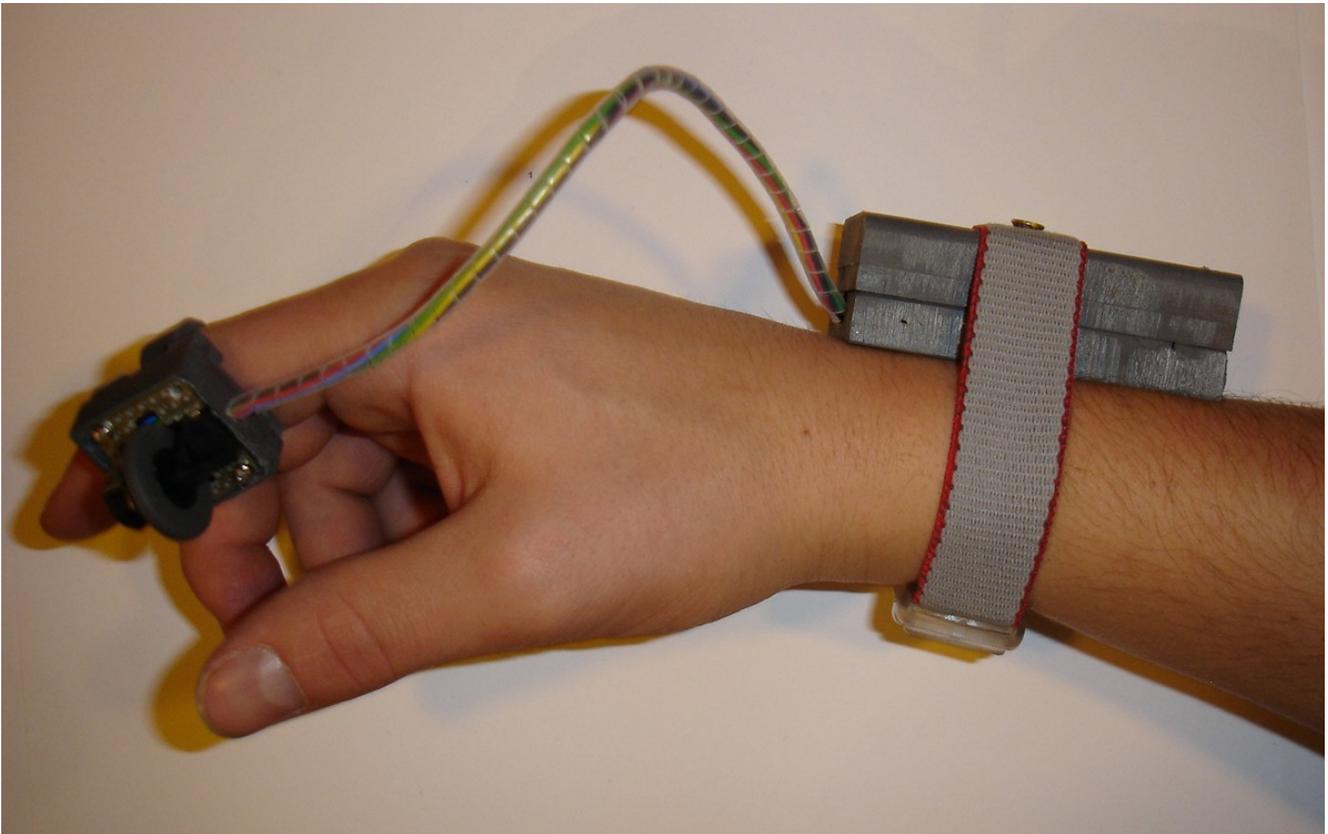


Figure 9: The final finger mounted Bluetooth Wheel

4. Menu Visualization

To test the above mentioned input device, a demo program had to be implemented. Hence I had to choose a programming language which makes it easy to create three dimensional objects and modify them. This brought me to Java3D which matches this specification perfectly, by using a scene graph model to manage and modify objects in a scene. Also creation and loading objects from other programs is quite simple.

4.1 Menu Structure

To make creating and modifying the menu structure easy, the structure is kept in a XML file. An example XML file is printed in Figure 10.

```
<Menustructure>
  <Item id="1" Title="Menu0">
    <Item id="2" Title="Menu0_0"/>
    <Item id="3" Title="Menu0_1"/>
    <Item id="9999" Titel=" ----Back----"/>
    <Item id="9998" Titel=" ----Exit----"/>
  </Item>
  <Item id="4" title="Menu1">
  </Item>
  <Item id="9998" Titel=" ----Exit----"/>
</Menustructure>
```

Figure 10: Sample XML File

As you can see, each menu item has an ID and a title. By the ID the item is recognized by the program. The title is the caption the menu item will get in the visualization. The IDs 9999 and 9998 are reserved for the “Back” and “Exit” buttons.

Using the DOM parser Java offers, the XML file is parsed and stored in a tree data structure. The tree for the file printed above should look like the one in Figure 11. Parsing the file and storing its data is done by “FileManager.java”. The tree data structure is implemented in

“XMLNode.java” and “XMLTree.java”.

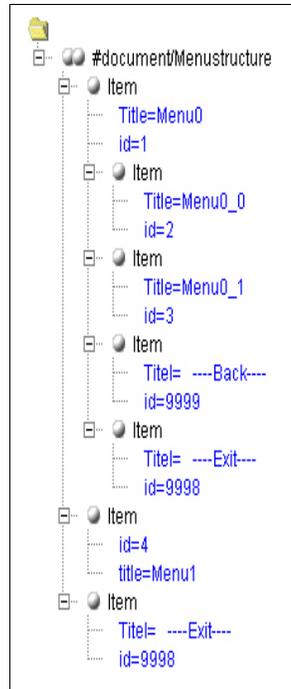


Figure 11: XMLTree structure

4.2 Menu Visualization demos

In the next few paragraphs I will take a look on the actual menu visualisation demos and how they are made.

Generally all four demos consist of the same components. Each demo has the two main classes called “<name>Menu” and “BG<name>”. <name> stands for the different visualisation names like “Roll”, “Classic”, “Slides” and “Ellipse”.

The class “<name>Menu” sets up a Java3D Scene by defining general scale, translations, lights, color of lights and some other scene properties. It provides a Canvas3D object to make the scene visible in a Java swing application. It implements a MouseListener and a MouseWheelListener to process the mouse events coming from the input device and react to

the input in the right way.

In this class also various “BG<name>” Instances are created, one for each submenu, based on the XML tree mentioned before. This is because “BG<name>” encapsulates the visualization of the single menu levels. It is a Java3D TransformGroup and is added to the scene graph in “<name>Menu”.

The menu items are created in a method called “createMenuItems”. It contains the alignments to position the menu items at the right place. When the wheel on the input device is rotated the methods nextItem and prevItem are called. In these Methods the next menu Item is selected and highlighted.

If a visualization needs a smooth transition between menu items, the transition is made in a timed thread.

The only parts that differ from visualization to visualization is the thread and the method “createMenuItems”, because these are the only parts that contain geometrical information. All other parts are the same in each visualization.

The single menu items themselves are created in a subclass called “BMenuItem” and is a TransformGroup. A menu item is just a box with a label in a specific colour. Several menu items are instantiated by “createMenuItems” for each menu item in a menu level.

In Figure 12 you can see a class diagram for the “Roll Menu” visualization. As mentioned above, the other visualization variants are implemented exactly the same way.

4.2.1. The “Roll Menu”

The idea of implementing a visualization like the one in Figure 13 came to my mind while surfing the Internet. I had a look at the site of TU Dresden and saw their sitemap. It looked quite similar to my implementation, but of course with a different interaction technique.

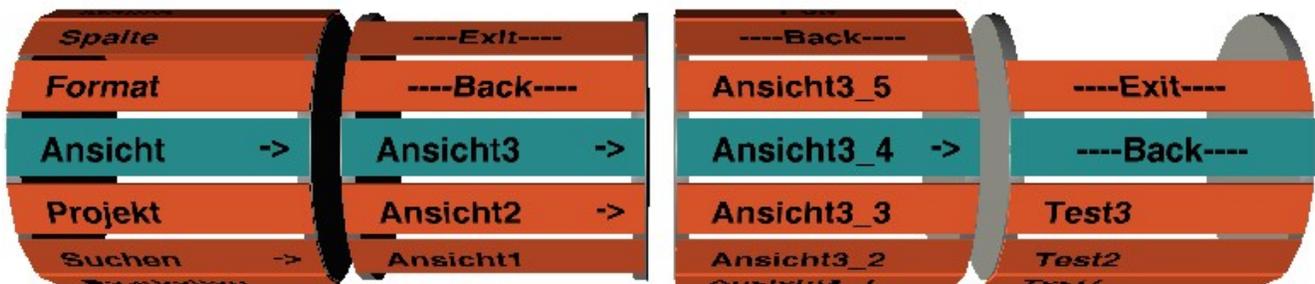


Figure 13: The "RollMenu"

As you can see on the picture, the demo consists of three dimensional rolls for each menu level. When the wheel on the input device is rotated, the roll on the active menu level is rolling up or down. The green color indicates the menu item as selected. Each wheel tick moves the marker one item further or backward. By clicking the wheel the selected menu item is chosen and a new roll appears right next to the other, representing the items submenu. If there are no submenus available, the item is just selected and an arbitrary action can follow. If you want to go up one level, you can either select the “Back” menu item or, if this option is switched on, move the wheel up or down while holding the button down. You have the same options if you want to leave the menu completely.

To intensify the 3D impression and to be able to follow a specific menu item when rotating fast, it is crucial to have a smooth rotation. As mentioned above, this is done by a timed thread. Because the actual selection can be faster than the timed rotation, the rotation follows the selection. Which means that you could have made a selection but the the roll is still rotating.

The “Roll Menu” visualization is probably the most intuitive variant, because if you imagine the wheel on the finger as a roll and rotate it, the roll on the screen just does exactly that rotation.

But if it is the best or not, will tell us the evaluation, which is discussed later.

4.2.2. The “Classic Menu”

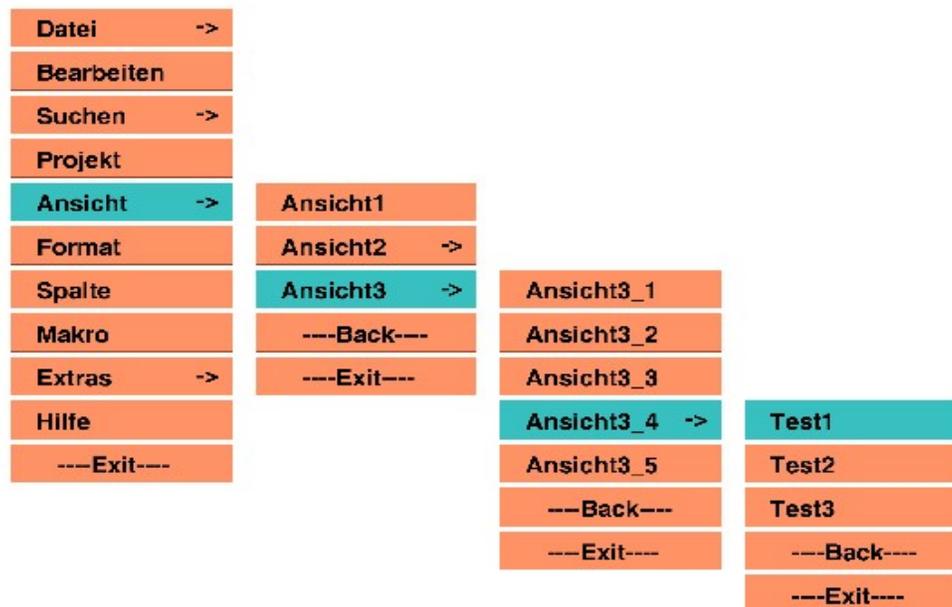


Figure 14: The "Classic Menu"

The most classic menu visualization is the default menu almost all window based operating systems offer. According to those menus, I implemented a variant using Java3D, but the visualization itself is two dimensional. A screenshot of this variant is shown on Figure 14. By moving the wheel on the input device up and down, the green selection on the screen moves up and down. Again, by clicking the wheel the selected Item is chosen and a submenu appears, just like in Windows. If one wants to go one level up, one can either select the “Back” menu item or, if this option is switched on, move the wheel up or down while holding the button down. One has the same options if he wants to leave the menu completely. Perhaps this will be the most popular variant, because everybody knows the menu structure and is used to work with that kind of menus. But as I said before, this will turn out in the evaluation part.

4.2.3. The “Slide Menu”

The biggest disadvantage all other menu visualizations have, is the big area they occlude in the field of view. This made me think about the visualization in Figure 15.

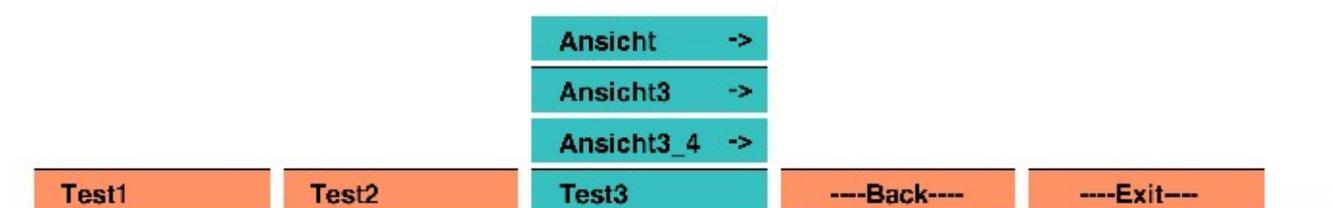


Figure 15: The "Slide Menu"

Here, the whole menu is located at the bottom of the screen. The menu items are ordered in a horizontal line. When the wheel is being rotated, the menu items move left or right. The selection always stays in the middle of the screen.

When an item is selected, the selected Item moves up and the new menu level appears right below it - again at the bottom of the screen. If one wants to go one level up, one can either select the “Back” menu item or, if this option is switched on, move the wheel up or down while holding the button down. One has the same options if he want to leave the menu completely. In this visualization it is also very important to have a smooth transition between the menu items when moving. If not, one cannot follow the menu items so that it is very uncomfortable to work with it.

4.2.4. The “Ellipse Menu”

The last menu visualization I implemented is an elliptic menu, which might be known from some DVD-Players or the Xbox. Again, this is a three dimensional visualization, as you can see in Figure 16. The object in the middle of the scene has no real use, but improves the three dimensional impression.

As you can see, the menu items are ordered in an ellipse. When being rotated by the input device, the ellipse is also rotating, with the menu items always facing the user.

The smooth transition between the menu items make it easy to follow the items and comfortable to use.

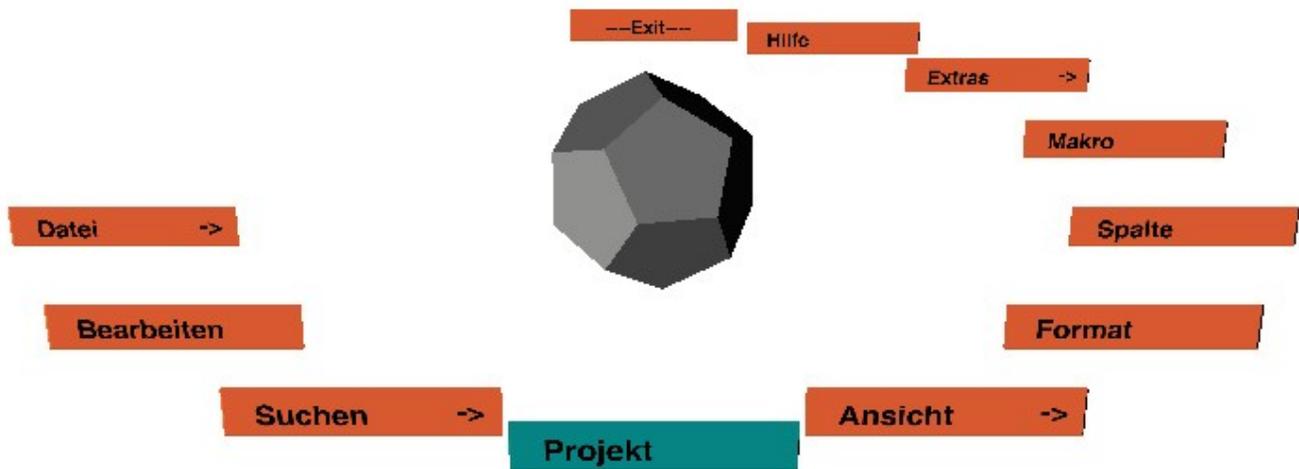


Figure 16: The "Ellipse Menu"

When an item is being selected, the old menu level disappears and the submenu appears. Again the items are ordered around the object in the middle. As in all other visualizations, if one wants to go one menu level up, one can either select the "Back" menu item or, if this option is switched on, move the wheel up or down while holding the button down. One has the same options if he want to leave the menu completely.

5. Evaluation

In this part, we will discuss how people rate the navigation wheel and the menu visualizations and we will see how good people can interact with them by measuring selection times and faults, a group of people made. First I will introduce the test procedure which is then followed by the actual results these tests offered.

A group of twelve male people (22 to 30 years old) tested the wheel in conjunction with the menu visualizations. The “within-subject” design was chosen, which means that all test persons test each of the menu visualizations.

The testing procedure is quite simple: For each menu visualization, a test person has to navigate to given menu items and select them using the Bluetooth wheel. The items the test person had to find were the following:

- 1.) Löwenzahn
- 2.) Adler
- 3.) Tanne
- 4.) Mercedes E-Klasse
- 5.) Peking-Ente
- 6.) Baseball

The first two items in this list are for testing purposes, which means that the results, these two items (Löwenzahn, Adler) offer, are not evaluated.

While the test person tries to find the menu items, the program logs important information about the user’s behavior: These are the time a user needs to find the correct menu item and the faults the user makes on the way to the menu item, like selecting a wrong item or navigating in the wrong menu branch.

After each menu visualization, the test person has to fill in a NASA-TLX questionnaire which tries to gather information about the user’s mental, physical and temporal demand, the user’s effort, his performance and his frustration level. For this purpose, the test person has to rate for each criteria on a scale from 0 to 100. An example questionnaire is given in Figure 17 on the next page. In this example the test person has to mark his impression on the scale below

the question.

Test 1: RollMenu

Geistige Anforderungen

Wie viel geistige Anstrengung war bei der Informationsaufnahme und -verarbeitung erforderlich (z.B. Denken, Entscheiden, Erinnern, Hinsehen, Suchen...)?

einfache Aufgabe | _____ | schwierige Aufgabe

Körperliche Anforderungen

Wie viel körperliche Aktivität war erforderlich? War die Aufgabe leicht oder schwer, einfach oder anstrengend, erholsam oder mühselig?

sehr wenig Bewegung | _____ | sehr viel Bewegung

Zeitliche Anforderungen

Wie viel Zeitdruck empfanden Sie hinsichtlich der Häufigkeit oder dem Takt, mit dem die Aufgaben bzw. die Aufgabenelemente auftraten? War die Abfolge langsam und geruhsam oder schnell und hektisch?

gering | _____ | hoch

Aufgabenerfüllung

Wie erfolgreich haben Sie Ihrer Meinung nach die gestellte Aufgabe erfüllt?

perfekt erfüllt | _____ | sehr unzufrieden

Anstrengung

Wie sehr mussten Sie sich insgesamt anstrengen, um Ihren Grad an Aufgabenerfüllung zu erreichen?

keine Anstrengung | _____ | sehr hohe Anstrengung

Frustration

Wie unsicher, entmutigt, irritiert, gestresst und verärgert fühlten Sie sich während der Aufgabe?

stresst mich überhaupt nicht | _____ | stresst mich sehr

Figure 17: A sample NASA-TLX questionnaire

By evaluating these questionnaires one can calculate an Overall Workload Index (OWI) which can then be compared for each menu visualization.

To eliminate the influence of the order of the tests, I permuted the test order according to the following Table:

Person	Permutation
1	RollMenu – ClassicMenu – SlideMenu – EllipseMenu
2	ClassicMenu – RollMenu – SlideMenu – EllipseMenu
3	EllipseMenu – ClassicMenu – SlideMenu – RollMenu
4	SlideMenu – EllipseMenu – ClassicMenu – RollMenu
5	RollMenu – EllipseMenu – SlideMenu – ClassicMenu
6	EllipseMenu – RollMenu – ClassicMenu – SlideMenu
7	EllipseMenu – SlideMenu – RollMenu – ClassicMenu
8	ClassicMenu – RollMenu – EllipseMenu – SlideMenu
9	EllipseMenu – SlideMenu – ClassicMenu – RollMenu
10	ClassicMenu – SlideMenu – EllipseMenu – RollMenu
11	EllipseMenu – ClassicMenu – RollMenu – SlideMenu
12	ClassicMenu – SlideMenu – RollMenu – EllipseMenu

Now we know how the tests are made, so we'll come to the test results.

We will start with the measured data. As mentioned before, time and errors the test person makes are logged for each menu visualization. The average selection times for each menu visualization are illustrated in Figure 18. As you can see, there are two values for each visualization. The red bar just contains the measured times for the logged Elements "Tanne", "Mercedes E-Klasse", "Peking-Ente" and "Baseball". The blue bar represents the corresponding variances.

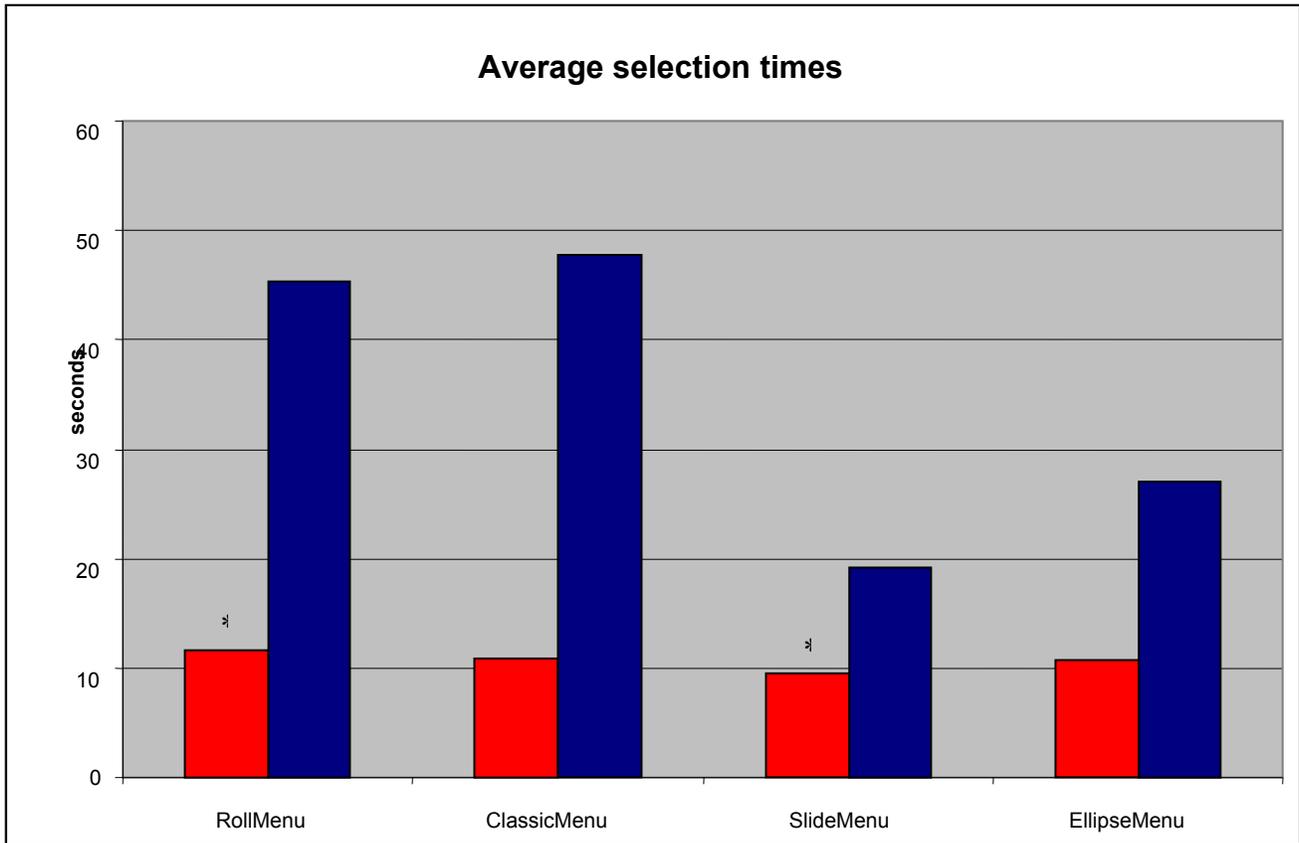


Figure 18 Average time for selection

To see if there is a significant difference between the selection times of the menu visualizations, one can calculate the ANOVA in Excel. The ANOVA calculation of the selection times without the test Elements is printed below:

Anova:

SUMMARY

Groups	Count	Sum	Average	Variance
RollMenu	48	559,956	11,66575	45,31238666
ClassicMenu	48	523,043	10,89672917	47,75509263
SlideMenu	48	459,496	9,572833333	19,18985372
EllipseMenu	48	513,78	10,70375	27,11483913

ANOVA

Source of Variati-	Squaresum	Deg. Of freedom	Avg. Squaresum	(F-value)	P-value	critical F-va-
Between groups	107,5926103	3	35,86420344	1,02930744	0,380814366	2,652645628
Within groups	6550,49209	188	34,84304303			
Total	6658,0847	191				

The more difference is between the selection times of the menu visualizations, the more differ the F-value and the critical F-value in the table above.

As we can see, these values differ, but not very much, so the difference between the visualizations is not very big. So when we take a look on Figure 18, we can figure out, that the SlideMenu has the lowest and the RollMenu the highest average selection times. These two Elements are marked with a "*" in Figure 18. The other two visualizations almost don't show any difference. This means, that it doesn't matter which of these two menu visualization is chosen. They have equal selection times. Within the other two visualizations, the SlideMenu has to be preferred, because it has lower selection times.

The same calculations were done with the errors made by the test persons. Figure 19 shows the average errors (red) with their corresponding variances (blue) for each menu visualization:

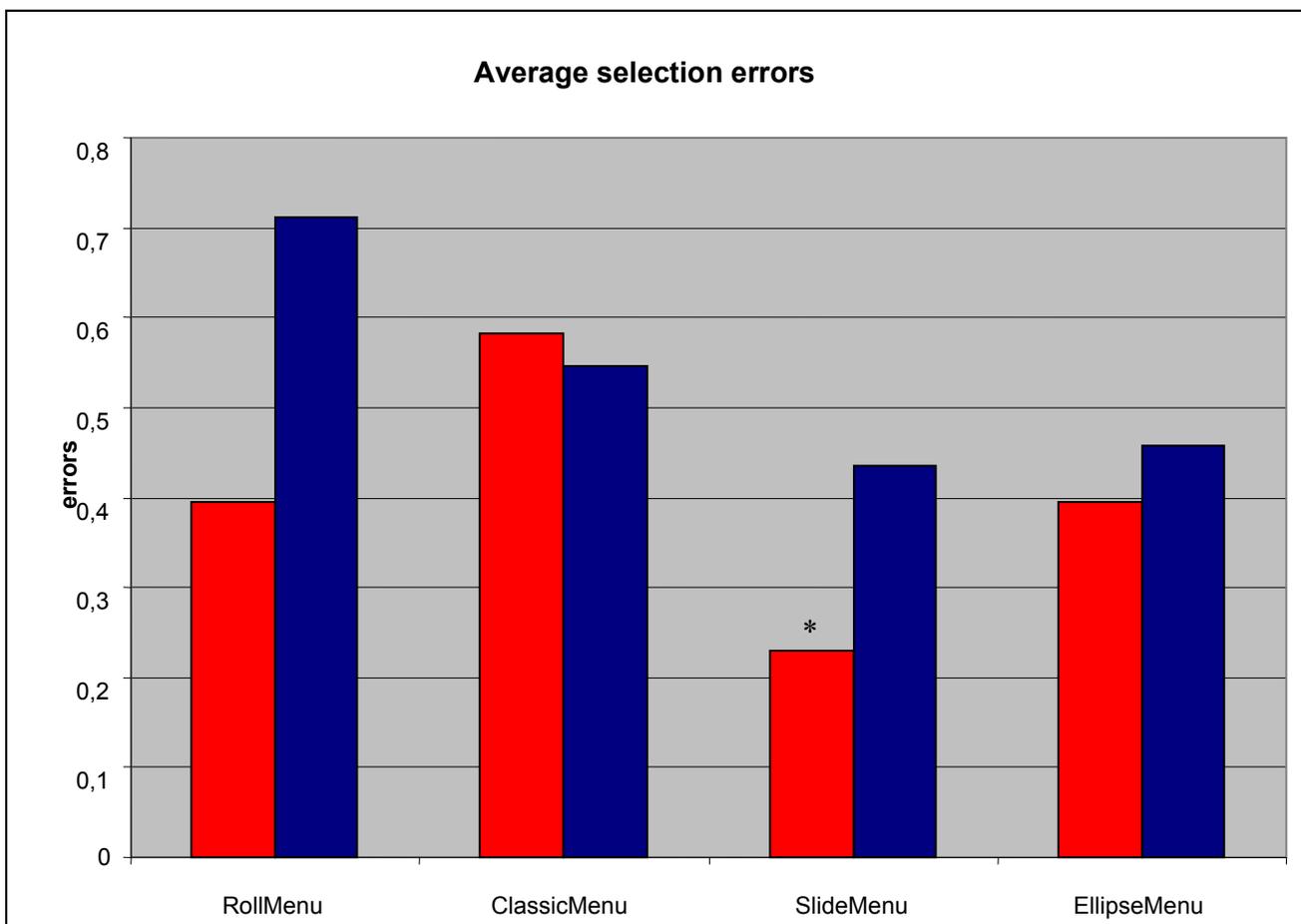


Figure 19 Average selection errors

The corresponding ANOVA calculation is printed in the following table:

Anova:

SUMMARY

<i>Groups</i>	<i>Count</i>	<i>Sum</i>	<i>Average</i>	<i>Variance</i>
RollMenu	48	19	0,3958333333	0,712322695
ClassicMenu	48	28	0,5833333333	0,546099291
SlideMenu	48	11	0,229166667	0,43572695
EllipseMenu	48	19	0,3958333333	0,457003546

ANOVA

<i>Source of Variati-</i>	<i>Squaresum</i>	<i>Deg. Of freedom</i>	<i>Avg. Squaresum</i>	<i>(F-value)</i>	<i>P-value</i>	<i>Critical F-va-</i>
Between groups	3,015625	3	1,005208333	1,869153101	0,136260591	2,652645628
Within groups	101,1041667	188	0,537788121			
Total	104,1197917	191				

Again, we compare the F-value with the critical F-value and can see, that there is a difference. The average selection errors can also be compared in Figure 19. As one can see, most errors were made in the ClassicMenu. Surprisingly, the fewest errors were made with the SlideMenu (marked with a "*" in Figure 19), although most people commented this visualization variant as not well arranged. Another comment many test persons spoke out was, that they made the errors because the wheel could be rotated too soft and they missed their target because of that. So this lets me come to the result, that the biggest amount of errors were made because of the wheel but not because of the complexity of the visualization. Perhaps the last step can give us more information to choose the best menu visualization. So we will evaluate the Overall Workload Index mentioned in the beginning. The OWI (red) and its variances (blue) for the menu visualizations are printed in Figure 20 on the next page, the ANOVA calculation is printed below the figure.

As illustrated in the figure the test persons had the most workload with the RollMenu visualization. The other visualization are almost equal, but the lowest workload was achieved with the ClassicMenu. Probably this is because this visualization is the most familiar variant.

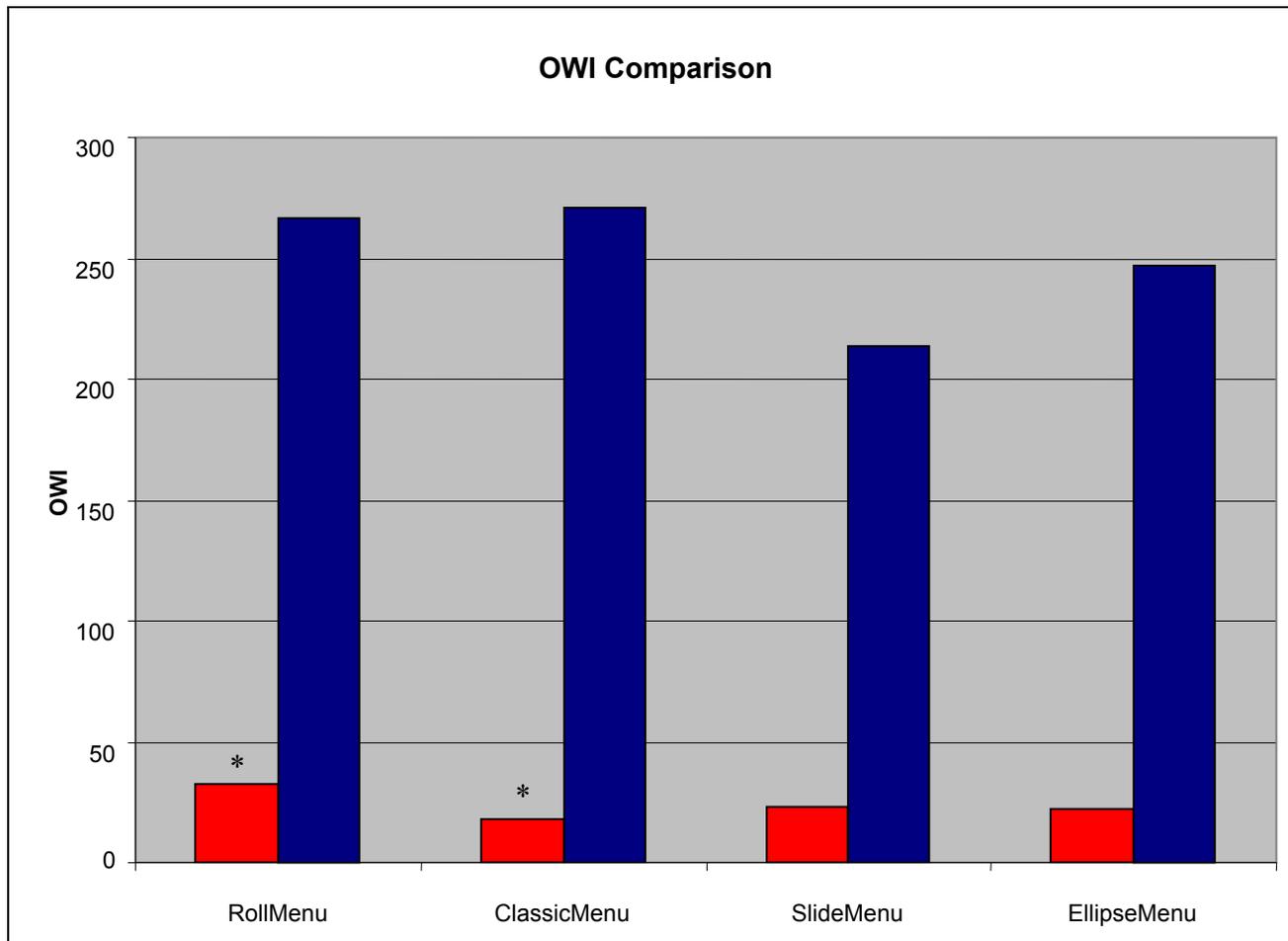


Figure 20 OWI Comparison

Anova:

SUMMARY

Groups	Count	Sum	Average	Variance
RollMenu	12	394,166667	32,84722222	266,4641204
ClassicMenu	12	217,2916667	18,10763889	270,7817103
SlideMenu	12	274,7916667	22,89930556	213,6478719
EllipseMenu	12	263,9583333	21,99652778	247,1627078

ANOVA

Source of Variati-	Squaresum	Deg. Of freedom	Avg. Squaresum	(F-value)	P-value	Critical F-va-
Between groups	1418,5574	3	472,8524667	1,89509315	0,144344452	2,816465827
Within groups	10978,62052	44	249,5141026			
Total	12397,17792	47				

As a final result I can say, that the SlideMenu visualization might be the best choice, to achieve good selection times, low error rates and a relatively low workload. Although the ClassicMenu had a little bit higher workload, I would rank it on place two, because error rate and selection times were much worse than with the SlideMenu.

This brings me to the generalization, that two-dimensional and one-dimensional menu visualizations lead to a better performance than three-dimensional ones. The reason for this might be, that in three-dimensional menu visualizations a lot of animation takes place, and confuses the user. This also matches some comments of the test persons.

6. Conclusion

In this document I have shown, how I developed a mobile input device for menu navigation issues. The result was the finger mounted Bluetooth wheel described in Chapter 3.

After constructing the wheel, four menu visualizations with one-, two-, and three-dimensional designs were implemented (Chapter 4) to be able to test the wheel in concerns of navigation times, error rates and workload, which is explained in Chapter 5.

As a final result, I can say, that the general idea of such a navigation wheel is a quite good idea. But for regular use, the components the wheel consists of, have to be of higher quality. For example the wheel itself can be rotated to easily, so that people have difficulties to select specific menu items.

As a second result of my work I can say that one- and two-dimensional menu visualizations are the better choice. They combine low error rates with good selection times and low workload.

Doing this project was a lot of fun, because it combined software programming with hardware developing, which is quite an interesting task. Because the evaluation done in Chapter 5 was the first I made, this was the task I learned the most.