CS3106: Practical 2 - Report

Matric Number: 170002815

18th November 2019

Overview

1 Question 1

The table described in the specification for Question 1 can be seen in *Figure 1* below, and the spreadsheet used to generate the table can be found in *Sheet1* of *Practical2.xlxs* contained in the same directory as this report. *Table 1* in *Figure 1* represents the data provided in the specification for different keyboards, while *Table 2* and *Table 3* represent the provided data for tablet pens and speech recognition systems respectively.

Results	Table 1	Table 2	Table 3
SSerror	80	72	596
SStotal	142.5	112	1440.666667
SSeffect	62.5	40	844.6666667
***n* Participants**	10	10	21
***m* Groups**	2	2	3
dferror	8	8	18
dfeffect	1	1	2
MSerror	10	9	33.11111111
MSeffect	62.5	40	422.3333333
α Confidence Level	0.05	0.1	0.01
F-ratio	6.25	4.444444444	12.75503356
Critical value	5.317655072	3.457918904	6.012904835
Significant?	TRUE	TRUE	TRUE
Table 1 Reporting	The QWERTY keyboard resulted in fewer average errors per unit time than the QuickPath keyboard (12 compared to 17 respectively). We assumed these errors were normally distributed. Analysis of variance at a significance level of a = 0.05 showed that this difference was statistically significant (F8,1 = 6.25, p < 0.05).		
Table 2 Reporting	The SurePen tablet pen resulted in fewer average errors per unit time than the QuickPen tablet pen (7 compared to 11 respectively). We assumed these errors were normally distributed. Analysis of variance at a significance level of a = 0.05 showed that this difference was statistically insignificant (F8,1 = 4.44 (3sf), p > 0.05); however analysis of variance at a significant level of a = 0.1 showed that this difference is statistically significant (F8,1 = 4.44 (3sf), p < 0.1).		
Table 3 Reporting	Siri resulted in fewer average errors per unit time than Alexa and Google Home (34 compared to 38 and 49 respectively). We assumed these errors were normally distributed. Analysis of variance at a significance level of a = 0.001 showed that this difference was statistically significant (F18,2 = 12.8 (3sf), p < 0.01).		

Figure 1: Table for Question 1

2 Question 2

2.1 Four factors to consider when deciding on whether to implement a linear menu or a pie menu

2.1.1 The number of entries in each level of a menu

I believe this factor is important to consider, as a large number of entries in a single level of a pie menu can seriously decrease its usability, however the effect is not quite as severe for a linear menu.

Figure 2 shows a pie menu with 16 entries, which is not unusually large, however it can be seen that the menu is already quite difficult to read. None of the text lies horizontal, which breaks consistency with almost every other form of notation users would otherwise encounter in their every-day lives, creating extra cognitive load to rotate and parse the text. Since each piece of text is also rotated to a different angle, it may be difficult for users to read multiple options at once. Although the text for some options (e.g. "Option 4") could be rotated to be perfectly horizontal, this would create internal inconsistency and decrease the aesthetic appeal of the menu. Contrast this to *Figure 3*, which shows a linear menu with 16 entries, all of which are displayed horizontally and require minimal effort to read.

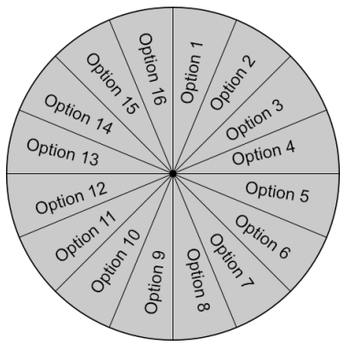

Figure 2: A Pie Menu with 16 Entries.

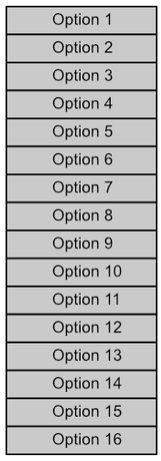

Figure 3: A Linear Menu with 16 Entries.

Another large problem for a pie menu with many entries in a single level is the lack of an initial focus point for the user. Some users may look to the top as if it were a clock, however some users may sub-consciously look towards the middle or right hand side of the menu, as text information is rarely ever encountered in this format. Not having a clear initial point of focus causes problems when implementing an order for the menu entries (e.g. alphanumeric), or sectioning out the menu with dividers. Once again, contrast this to the linear menu displayed in *Figure 3*, which has a clear initial focus point (the top of the menu), making it easy to implement a natural ordering and sectioning.

A real-world example of a linear menu with a large number of entries can be seen in *Figure 4*, which displays Affinity Designer's [1] "View" menu containing a total of 28 entries; the menu content is sectioned clearly, easy to read, and is sorted (presumably) by frequency of use.

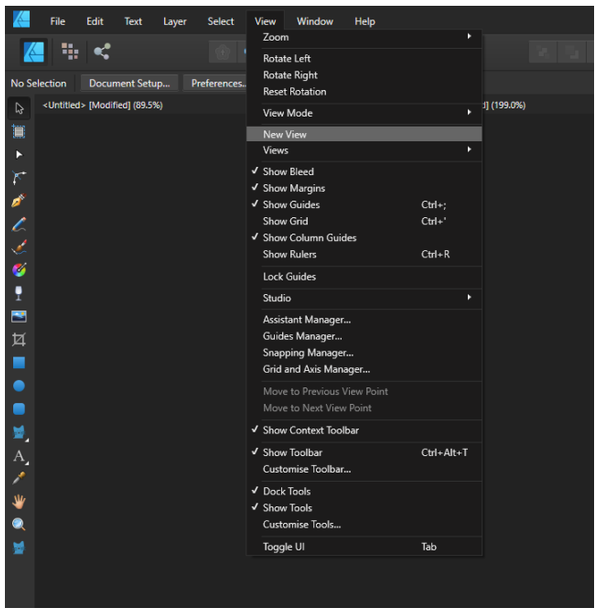

Figure 4: Affinity Designer's [1] "View" Menu Containing 28 Entries.

To compare the linear menu shown in *Figure 4* to a pie menu, I decided to create one with the same entries and also the same pixel area. To ensure the areas were the same, I used my graphics design software [1] to measure the original menu, and then used the following mathematics to determine the radius needed for the new menu.

$$Length\ of\ linear\ menu\ according\ to\ design\ software = 291px$$
$$Width\ of\ linear\ menu\ according\ to\ design\ software = 644px$$

$$Area\ of\ linear\ menu = 291px \times 644px = 187404px^2$$
$$Area\ of\ pie\ menu = \pi r^2 = 187404px^2,\ where\ r\ is\ the\ radius\ of\ the\ menu$$
$$r = \sqrt{\frac{187404px^2}{\pi}} = 244.24px\ (2d.p.)$$

The equal-area pie menu and linear menu containing the same entries can be seen on the left and right hand side of *Figure 5* respectively. Aesthetics aside, since the design process was relatively short, the pie menu is much more difficult to read than linear menu, with many of the points discussed earlier in this section being represented clearly.

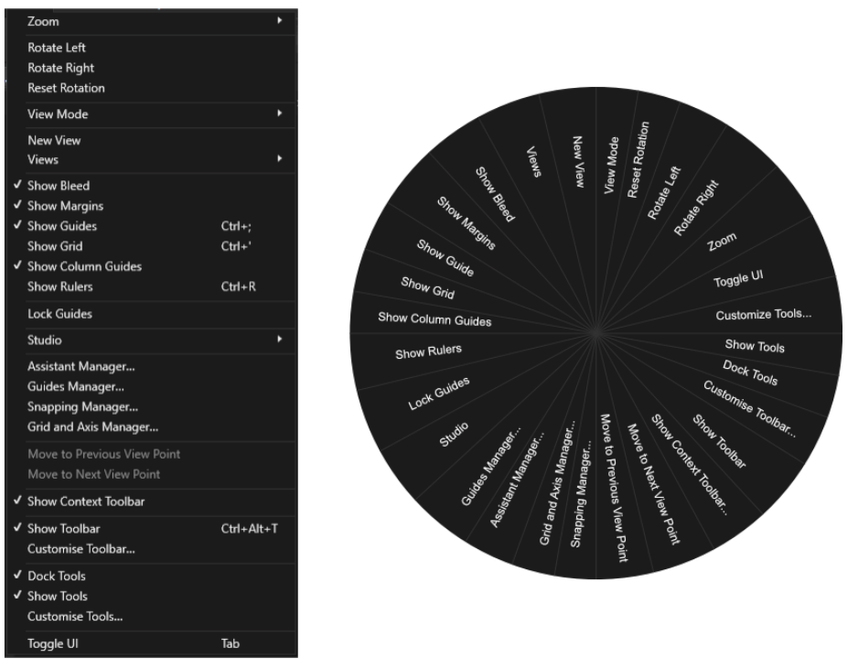

Figure 5: *Figure 4*'s Menu (Left) Compared to a Same-Area Pie Menu (Right).

One final point to consider for this factor is the availability of overflow scroll in a linear menu in comparison to a pie menu. If there are too many entries to be displayed effectively in a single menu, an overflow can be created that can be scrolled into view - an example of this implemented in a linear menu is shown in *Figure 6*. Although a form of overflow scroll could be implemented for a pie menu, it would be much less intuitive, and also would require much more development, as many UI frameworks already offer linear menu overflow scrolling (e.g. HTML).

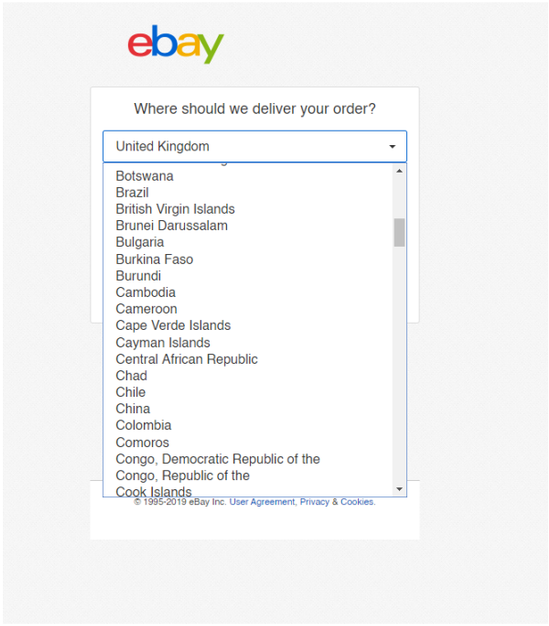

Figure 6: eBay's [2] Country Selection Menu Demonstrating Overflow Scroll.

To conclude the consideration of this factor, if there are a relatively large number of entries in the menu being implemented (approx. more than 10), then I would recommend using a linear menu, as it reduces the user's cognitive load, and also creates a much better consistency with other text base information sources.

2.1.2 The depth of a menu (i.e. the number of levels)

This factor will usually not need to be considered thoroughly when deciding between a linear and a pie menu, however if there are a relatively large number of levels in a menu, then this factor could greatly influence the decision.

Linear menus with many levels are fairly unfriendly to users, as shown by UserTesting's YouTube video named "Multi Level Nav" [3], however I believe there are situations where they are superior to a pie menu with the same number of levels. The main situation I believe this is the case, is when the space a menu can take up is relatively limited - this will occur when developing menus for any type of screen, particularly small ones, such as mobile phones or tablets. The problem with pie menus in this scenario is that they need to be able to expand in every direction to allow for multi-level traversal; however linear menus only need to expand in at most two directions.

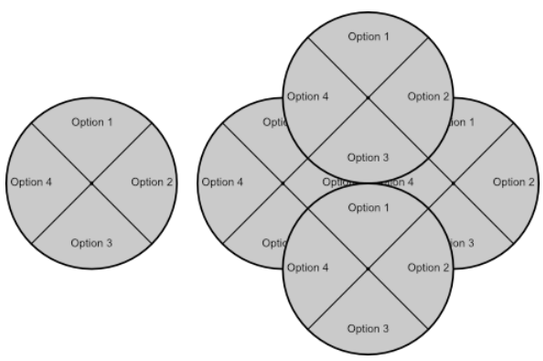

Figure 7: Expansion Required by a Pie Menu.

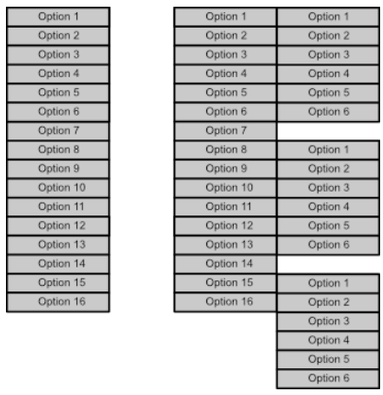

Figure 8: Possible Expansion Required by a Linear Menu.

Figure 7 shows the expansion needed for a pie menu with the depth of 2 levels, while *Figure 8* shows the possible expansion needed for a linear menu with the same number of levels. By completing the mathematics shown below, it can be seen that the expansion needed by a pie menu is twice that for the expansion needed by a linear menu, assuming that the linear menu only expands in one direction and all the entries are the same size, which is usually a fair approximation. This expansion effect occurs for every level, meaning that a pie menu will take up the available space sooner than a linear menu with the same number of levels.

$$Original\ Pie\ Menu\ Radius = r$$
$$Expanded\ Pie\ Menu\ Radius = 2r$$

$$Original\ Pie\ Menu\ Area = A_p = \pi r^2$$
$$Expanded\ Pie\ Menu\ Area = \pi(2r)^2 = 4\pi r^2 = 4 \times A_p$$

$$Original\ Linear\ Menu\ Height = h = Expanded\ Linear\ Menu\ Height\ (Assumption)$$
$$Original\ Linear\ Menu\ Width = w$$
$$Expanded\ Linear\ Menu\ Width = 2w\ (Assuming\ all\ entries\ same\ size)$$

$$Original\ Linear\ Menu\ Area = A_l = w \times h$$
$$Expanded\ Linear\ Menu\ Area = 2w \times h = 2 \times A_l$$

If a menu is considerately designed, the number of levels should be minimised, however in the situation where there are a relatively large number of levels (approx. greater than 2), I would recommend a linear menu.

2.1.3 Menu placement

This factor relates closely to the required depth of a menu, however should still be considered when deciding between a linear menu and a pie menu. Usually a menu is implemented to fit with the rest of an interface, rather than the other way around, meaning the menu placement is predetermined before the choice between a linear and pie menu is made.

As discussed in *Section 2.1.2*, an expansive pie menu requires more space in every direction than an expansive linear menu, meaning if the menu placement is towards the edge or corner of an interface it would be sensible to implement a linear menu to avoid any overflow off the side(s) of the interface. If a circular menu is wanted for other reasons (e.g. UI design heuristics, as discussed in *Section 2.1.4*), the position of the entries can be altered to allow the pie menu to not overflow when placed in the corner of an interface - an example of this is shown in *Figure 9*.

Figure 9: Matt Sclarandis' Google Chrome Circular Menu Concept [4]

Although *Figure 9* shows a clever implementation of a pie menu for corner placement, it may not be considered a traditional pie menu. If a traditional pie menu and a linear menu are the only options, I would recommend using a linear menu if the required placement is near an edge or corner.

2.1.4 User interface design heuristics

An important factor to consider when implementing any component of a user interface, including a menu, is how well it relates to UI design heuristics, such as Nielsen's list of ten heuristics, or Schneiderman and Plaisant's list of eight golden rules. Sticking closely to heuristics such as these will increase the likelihood that the implemented component is user-friendly.

Examples of how design heuristics should be considered when choosing between a linear and a pie menu include using a linear menu if all other menus already implemented are linear, as this fits with Schneiderman and Plaisant's "Strive for consistency" heuristic. Another example is using a pie menu to fit with Nielsen's "Flexibility and efficiency of use" heuristic, as the button to open pie menu could be placed anywhere on a page, while a linear menu will usually have to be mounted to something to avoid breaking other good design practices. One final example may be using a linear menu on a website, as it is consistent with other menus across the web, fitting well with Nielsen's "Consistency and standards" heuristic.

There is no definitive answer to which style of menu fits better with all UI design heuristics, and therefore I would recommend considering this factor on a case-by-case basis.

2.2 An experiment to empirically compare pie menus and linear menus using between-subjects design

2.2.1 Hypothesis

There is no significant difference between time to use pie menus and linear menus.

2.2.2 Independent & Dependent Variables

Independent Variable(s): The type of menu used and the entry specified to navigate to.
Dependent Variable(s): The time taken to select the correct entry from opening the menu.

2.2.3 Setup

The experiment will be run in the same room, using the same PC, mouse, monitor, web browser, browser size & zoom level, and menu entries for every participant. Chair height, monitor height & brightness, and cursor speed may all be adjusted.

2.2.4 Participants

Selected participants must be able to see, and have full motor control of their muscles. Daily use of a computer is preferable to promote external validity.

2.2.5 Apparatus

A monitor, mouse, computer, desk and chair are required.

2.2.6 Material

Software to generate a linear menu and a pie menu from a set of entries is required.

2.2.7 Study Procedure

1. Explain the experiment to the participant.
2. Position the participant in the seat, and allow them to adjust the variables mentioned in *Setup*.
3. Setup the menu being used for the participant, using the linear menu for the first participant and alternating thereafter.
4. Allow the participant 120 seconds to explore the menu.
5. Ensure the menu is closed, then randomly generate an entry from the menu and show the participant the text for this entry for as long as they would like.
6. Get the participant to open the menu and navigate to the entry as fast as possible. Collect data on the time taken to complete these tasks, including the time to correct errors.
7. Repeat steps 6 and 7 until at least 20 timings are collected for each participant.
8. Repeat steps 1 through 8 with different participants.
9. Statistically test the hypothesis for this experiment using this data.

2.2.8 Skill Transfer & Balancing

No balancing is required, as skill transfer is eliminated by using between-subjects design.

[Word Count (Excluding Titles and Enumeration): 300 Words]

2.3 Validity concerns to the experiment proposed in *Section 2.2*

Many measures are taken to avoid issues with validity in the experiment described in *Section 2.2*, such as gathering users who use computers daily to promote external validity, however there are still some issues which could affect the validity of the experiment. The two most important threats to consider in my opinion are described below.

2.3.1 Participant Selection

I believe participant selection is a large threat to the internal validity of the experiment. Although some measures are described to match participants better to the general user of linear and pie menus to increase external validity, there are no steps taken to make sure the participant selection is random and without bias, creating a risk to internal validity.

The specification for participants is vague, with the definition of "able to see" in particular being very ambiguous. This could lead to experimenter bias and perhaps even judgemental sampling, as it takes the experimenter's opinion into account - both of these reduce the internal validity of the experiment.

To reduce the threat imposed by the described participant selection process, I would include a more technical specification of the requirements, such as "an eyesight of at least 20/30", and also state allowed sampling techniques, such as stratified sampling and cluster sampling.

2.3.2 Interaction Technique

The interaction technique used for this experiment is exclusively a computer mouse, however even at the time of writing, there are many more techniques which could be used to interact with a linear/pie menu, therefore reducing the experiment's relation to the real world, in turn reducing its external validity. Some different interaction techniques include a touchscreen, a track pad, and VR controllers.

To reduce the threat imposed by only using a mouse, I would suggest changing the objective of the experiment to be less vague, e.g. "an experiment to empirically compare two menu styles when interacted with using a computer mouse". Unfortunately this reduces the scope of the application of the result of the experiment.

An alternative method to reduce the described threat would be to include other interaction techniques, however this greatly increases the complexity of the experiment, as a new independent variable is introduced, and appropriate precautions need to be taken in response.

3 Question 3

3.1 Menu Design & Implementation

3.1.1 Design Process

To begin the design of my linear menus, I first completed some research. I explored Overleaf's site [5] to gather information on the design style currently being used - this information included colours, fonts, interactions and shapes, all of which I tried to incorporate into my menu designs to promote internal consistency with the rest of the site. To further my research, I decided to look at some examples on the internet to gather a general idea of how a linear and pie menu should look - the main site used for this was Dribbble [6].

After completing my research, I experimented with a few different designs for both types of menus, trying to incorporate as much of Overleaf's design style as possible. As the time allowed for the design & development process of this practical was fairly short, I decided to create some initial designs and tweak them as needed during the implementation process - you can see a few of my initial designs in *Figure 10* below.

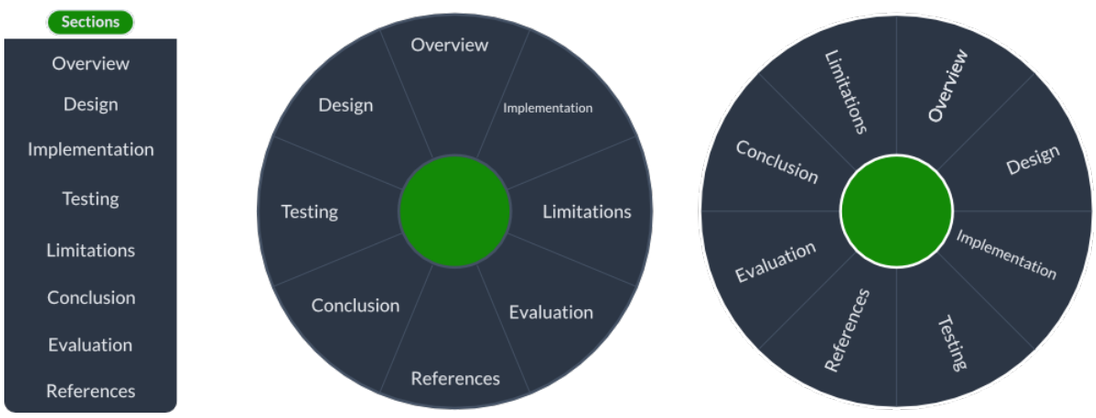

Figure 10: Initial Menu Designs

3.1.2 Implementation

To complete my implementation of my menus, I created an extension for Google Chrome, which can be found along with a guide to setup the extension (README.txt) in the *OverleafSectionMenus* directory in the same directory as this report. The extension uses JavaScript to inject new HTML elements and CSS styles into the page, as well as create interactivity for the elements - the JavaScript used to complete these processes can be found in *content.js*. I also used this JavaScript file to parse the current open document, however this is discussed further in the *Extensions* section of this report.

During my implementation, I decided to mount the linear menu in the toolbar above the editor, since this area was easy for the user to access and had nothing currently occupying the space. I made the decision for the pie menu to be unmounted and for the user to be able to move the menu around the screen freely, as this avoids the issues discussed in *Section 2.1.3* of this report. Interact.js was used to make the menu "draggable", as the process is otherwise very complicated and would take a significant amount of development to optimise [7].

Apart from the "draggable" functionality of my pie menu, I made it all from scratch, and therefore decided to limit the number of entries to 8; however the linear menu does not have a limit for how many entries it can hold, as it has scrollable overflow. NB: Overleaf dislikes any "draggable" objects, and therefore the pie menu is contained to the PDF viewer of the page, to break the fewest things, however the compile button is dysfunctional (although Ctrl + Enter works as usual).

A file named *OverleafMenus.mp4* can be found in the same directory as this report, and contains a video of both my linear and pie menus in action.

3.2 Experiment Data & Analysis

3.2.1 Timing & Error Collection

Upon completing my linear and pie (radial) menu implementations, I modified the JavaScript code to output the time taken to complete a selection to Google Chrome's console - this provided me an accurate and reliable method to gather timing data for my experiment.

Error data was not mentioned in my experiment design in *Section 2.2* of this report, as the specification did not require it, however the specification did require that error data was gathered during the running of the experiment. To gather this data, I decided to use participant's honesty in letting me know if they made an incorrect selection - I was also able to see if they had made the wrong choice while watching them complete the experiment. To incorporate errors into the timing data, I decided to include the time required to correct any errors in each time recorded, as this aligned properly with the experiment specification.

3.2.2 Participants

To select participants for this experiment, I unfortunately had to use convenience sampling, as I decided to run the experiment in my flat rather than the Computer Science building - this is a poor choice in hindsight. Although this does not disagree with the experiment specification given in *Section 2.2* of this report, it does influence the experiment's validity, as discussed in *Section 2.3.1*.

Due to the time constraints of this practical, only 6 participants were involved, which is a relatively small number, potentially further reducing the experiment's validity, however all participants matched the experiment's requirements of having sight and full motor control.

3.2.3 Data

Although, the experiment specification required 20 points of data, I decided to collect 40, as the time required to do so was fairly short, and this also increased the validity of my experiment. After collecting the data, I took a median value for each participant, because this reduces the influence of outliers, which I believe better represents the average use case of each menu. The number of errors made were normalised, to make sure the value was representing errors per single use for each participant.

The data collected can be viewed in *Sheet2* of *Practical2.xlsx*.

3.2.4 Analysis

To analyse the data collected from the experiment, I decided to use ANOVA with 2 groups and 6 participants on the median timing values and the number of errors separately - both processes and results can be seen in *Sheet3* of *Practical2.xlsx*.

The conclusion for the timing data is that although the linear menu resulted in a shorter average use time than the pie (radial) menu (937.3ms (1d.p.) compared to 1185.7ms (1d.p.)), analysis of variance at a significance level of $\alpha = 0.05$ showed that this difference was statistically insignificant ($P(F_{4,1} = 1.45(2d.p) > 0.05$). The data used is assumed to be normally distributed in this calculation. This conclusion agrees with the original hypothesis, and therefore there is no reason to reject it.

The conclusion for the error data is that although the linear menu resulted in fewer average errors per use than the pie (radial) menu (0 compared to 0.083 (3d.p.)), analysis of variance at a significance level of $\alpha = 0.05$ showed that this difference was statistically insignificant ($P(F_{4,1} = 1) > 0.05$). The data used in assumed to be normally distributed in this calculation.

3.2.5 Conclusion

Although the assumption of normally distributed data is a fair assumption for each ANOVA, as it is recorded from human performance, and the experiment was completed in a suitable environment and with a proper procedure, I believe that the results are most likely invalid, since there are very few participants, who were all selected using convenience selection. Therefore, although my experiment suggests there is no significant difference between time to use and number of errors of linear and pie menus, I believe this result should not be used to make any inferences about the population.

4 Extensions

To extend the basic specification of this practical, I decided to attempt to parse the LaTeX document currently open in Overleaf [5] in order to fill the menus. The practical specification suggested this was a challenging extension and it took me a lot of trial and error to proceed at all, and unfortunately I could not complete it.

The largest problem I encountered is that Overleaf does not create an element for each line in a document, but rather creates elements for only those visible by the user at the time of reading the website's DOM, meaning not all lines could be parsed without either the viewer being moved or faking the user viewing more lines - each have their own issues described below.

To move the viewer, I found adjusting the "scroll" value of the scroll bar changed the position of the viewer - this is already made use of when selecting an element in the menu. The problem with parsing the entire document using this method is that the viewer is only updated to match the scroll bar when the JavaScript I am running has finished its current task, including functions set to a timeout. This means that without a complicated method involving user input, the entire document cannot be parsed, and even if this method was implemented, user input would be needed to search for a section, making the search function redundant.

Faking the number of lines a user could view was a more difficult task. After many hours of tweaking different elements' CSS, I found a way to display all possible lines at once - set the height of the only element with class name "ace-editor-wrapper" to be very large (e.g. 10000000px), scroll to the top of the document, zoom out once then zoom back in. Although this method creates a line element for every line in a document, the scroll became dysfunctional after completing it, and therefore I could parse the entire document, but not scroll to a selected section, making this method redundant.

Although I was very determined to complete this extension, Overleaf's complexity got the better of me and my implemented functionality goes as far as parsing the sections which are shown in the viewer at the time of loading or refreshing (using the refresh buttons on the menus). I believe the only way to complete this extension would be if Overleaf created an API to query the document's data from the database, however this comes with its own issues, particularly in relation to data privacy.

Examples of the menus parsing sections from an Overleaf document are demonstrated in *OverleafMenus.mp4*.

5 Evaluation & Reflection

I believe I have been very successful in this practical, since I have completed the basic specification to a high standard, as well as completed part of a difficult extension. Reflecting on the learning outcomes of this practical, I feel much more confident with ANOVA, as well as the design and execution of experiments. I have also learned how to create a Chrome Extension from scratch, and have a greater understanding of how Overleaf's editor functions.

Given more time on this practical, I would enjoy contacting Overleaf about a way to parse the entire document and finishing my extension; given a chance to redo this practical, I would experiment using more participants and consider other ways to analyse my data (e.g. sign test).

References

[1] Affinity designer - professional graphic design software for desktop and ipad.
https://affinity.serif.com/en-gb/designer/.

[2] ebay.
https://www.ebay.co.uk/.

[3] Usertesting - multi level nav.
https://www.youtube.com/watch?v=q01PGEeyU8ofeature=emb$_{l}ogo$.

[4] Matt Sclarandis. Matt sclarandis' google chrome circular menu concept on dribble.
https://dribbble.com/shots/3458945-Google-Chrome-Circular-concept-Menu-for-incredibly-big-screens.

[5] Overleaf, online latex editor.
https://www.overleaf.com/.

[6] Discover the world's top designers creatives.
https://dribbble.com/.

[7] Javascript drag and drop, resizing, and multi-touch gestures for modern browsers (and also ie9).
https://interactjs.io/.