Ostfalia - Hochschule für angewandte Wissenschaften
Fakultät Informatik
Studiengang Informatik

Bachelorarbeit

Entwicklung von UI-Widgets zur numerischen Werteeingabe in Marking-Menus

eingereicht bei Prof. Dr. J. Weimar
j.weimar@ostfalia.de
Zweitprüfer S. Schneegans
simon.schneegans@dlr.de

von Hannes Kruse
Wiesengrund 1
31608 Marklohe
70451368

Wolfenbüttel, den 16. September 2019

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit mit dem Titel

Entwicklung von UI-Widgets zur numerischen Werteeingabe in Marking-Menus

selbstständig und ohne unerlaubte fremde Hilfe angefertigt, keine anderen als die angegebenen Quellen und Hilfsmittel verwendet und die den verwendeten Quellen und Hilfsmitteln wörtlich oder inhaltlich entnommenen Stellen als solche kenntlich gemacht habe.

_____

Hannes Kruse

Wolfenbüttel, den 16. September 2019

Zusammenfassung

Die vielfältig beschriebene Interaktionsform der Pie-Menüs bietet gegenüber linearen Menüstrukturen distinktive Bedienvorteile. Eine Integration dieser Menüs in aktuellen Computersystemen findet jedoch kaum statt. Der Funktionsumfang eingesetzter Pie-Menüs ist beschränkt auf die Ausführung einfacher Befehle, dasselbe gilt für die erweiternde Form der Marking-Menüs. Gängige Steuerelemente eines Benutzerinterface wurden für den Einsatz in diesen Menüs nicht konzipiert, eine Integration dieser Widgets fand nur teilweise bis gar nicht statt.

Diese Arbeit gibt einen Überblick über den Stand der Technik von Pie- bzw. Marking-Menüs, beschreibt gängige Steuerelemente einer Benutzeroberfläche und zeigt, wie diese in einem Marking-Menü integriert werden können.

Insbesondere das Element des Schiebereglers wird im Verlauf dieser Arbeit analysiert und aufgearbeitet. Das Ergebnis einer Pilotstudie zeigt unter fünf getesteten Prototypen die Präferenzen der Anwender. Aufbauend auf dieser Studie erfolgt eine Überarbeitung und Implementierung des bestbewerteten Prototypen in einem zuvor entwickelten Marking-Menü Framework.

Abstract

The interaction form of pie menus, which is described in many ways, offers distinctive operating advantages over linear menu structures. However, these menus are hardly integrated into current computer systems. The functional range of pie menus used is limited to the execution of simple commands, and the same applies to the extended form of marking menus. Common control elements of a user interface were not designed for the use in these menus, an integration of these widgets took place only partially or not at all.

This paper gives an overview of the state of the art of pie and marking menus, describes common controls of a user interface and shows how they can be integrated into a marking menu.

In particular, the element of the slider is analyzed and processed in the course of this work. The result of a pilot study shows the preferences of the users among five tested prototypes. Based on this study, the best-rated prototype is revised and implemented in a previously developed marking menu framework.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Bereits 1988 erwiesen Callahan u. a. in ihrem Beitrag „An empirical comparison of pie vs. linear menus“ [3], dass Pie-Menüs gegenüber linearer Menüstrukturen schneller zu navigieren sind und eine geringere Fehleranfälligkeit aufweisen.

Ein Blick auf aktuelle Computersysteme zeigt konträr, dass die Verbreitung von Pie-Menüs geringfügig ausfällt und diese wenig eingesetzt werden.

Nur einzelne Anwendungsgebiete setzen die vorteilhaften Pie-Menüs ein. Ihr Funktionsumfang ist aber meist auf die einfache Ausführung von Befehlen beschränkt. Die Integration gängiger Steuerelemente einer Benutzeroberfläche fand in der Folge teilweise bis gar nicht statt.

Dies soll der Ausgangspunkt der nachfolgenden Arbeit sein. Welche Steuerelemente können in einem Pie-Menü umgesetzt werden? Mit Fokus auf dem Element des Schiebereglers, zur Eingabe numerischer Werte, sollen die Fragen beantwortet werden: Wo liegt das Problem bei der Umsetzung von Schiebereglern in Pie-Menüs? Wie kann das Steuerelement des Schiebereglers sinnvoll in ein Pie-Menü integriert werden?

Für Beantwortung dieser Fragen soll das Steuerelement des Schiebereglers in einem modernen Marking-Menü umgesetzt werden. Das Marking-Menü ist eine Erweiterung des Pie-Menüs, dessen Definition 1993 auf Gordon Kurtenbach und Buxton [20] zurückgeführt werden kann. Erkenntnisse dieser Arbeit können neben dem Marking-Menü gleichermaßen auf das Pie-Menü angewendet werden.

1.2 Ziel der Arbeit

Das Ziel dieser Arbeit ist die Integration der Klasse des Schiebereglers in die Struktur des Marking-Menüs. Es soll eine Antwort auf die Frage gefunden werden, wie Schieberegler zur Eingabe numerischer Werte in einem Marking-Menü umgesetzt werden können und welche Anforderungen diese erfüllen müssen, um sinnvoll eingesetzt werden zu können.

Neben der Betrachtung des Schiebereglers soll als Grundlage eine Übersicht über den Stand der Technik von Pie- und Marking-Menüs gegeben werden. Betrachtet werden sollen die Bedienungsarten und Einsatzgebiete radialer Menüs sowie die standardisierten Widgets heutiger Benutzeroberflächen. Analysierte Widgets sollen ferner auf ihre Umsetzbarkeit in Pie-Menüs bewertet werden.

1.3 Eingrenzung des Forschungsfelds

Um eine klare Basis für die Beantwortung der aufgestellten Fragen dieser Arbeit zu schaffen, soll das Forschungsgebiet in seinem Umfang eingegrenzt werden.

Neben der Umsetzung einfacher Steuerelemente liegt der Fokus dieser Arbeit auf der Erforschung von Schiebereglern zur numerischen Werteeingabe in Pie- bzw. Marking-Menüs. Dies schließt die Betrachtung nominaler und ordinaler Schieberegler aus.

Die zu entwickelnden Schieberegler sollen darüber hinaus zusätzlich auf ihre Bedienbarkeit durch Toucheingaben am Computer bewertet werden.

1.4 Kapitelübersicht

Kapitel 2.: *Stand der Technik* beschreibt die Idee der Pie-Menüführung, den historischen Kontext aus dem Pie-Menüs hervorgegangen sind, sowie die speziellen Eigenschaften, die die performante Bedienung ermöglichen. Ferner werden die Anwendungsfälle und Bedienungsarten dieser Menüs erläutert. Kapitel 2.4.: *User Interface Widgets* behandelt die gängigen Widgets einer Benutzeroberfläche und analysiert diese in Hinblick auf die Integration in Pie-Menüs. Nachfolgend wird das Marking-Menü erläutert, das eine Erweiterung des Pie-Menüs ist und als Grundlage aller Implementierungen dieser Arbeit dient. Beschrieben wird darüber hinaus im Detail die Widgetklasse der Schieberegler sowie einer beispielhaften Integration dieser in ein Marking-Menü.

Kapitel 3.: *Marking-Menü Framework* verschafft eine Übersicht über das zuvor entwickelte Marking-Menü Framework. Dieses Kapitel beinhaltet Informationen über die Struktur, die Elemente und das Layout des Marking-Menüs.

Kapitel 4.: *Prototyp Implementierung* stellt in einer Anforderungsanalyse und Bewertung verschiedene Anforderungen an zu entwickelnde Schieberegler. Anhand der klassifizierten und gewerteten Anforderungen werden vier entwickelte Prototypen beschrieben. In einer anschließenden Pilotstudie werden die Schieberegler durch Anwender getestet und in den Kategorien Verständlichkeit, Spaß und Performance bewertet. Die Auswertung der Ergebnisse gibt Hinweise auf implementierenswerte Schieberegler. Nutzerkommentare der Studie gehen in die Überarbeitung des vielversprechendsten Prototypen ein. Das Kapitel Kapitel 4.4.: *Umsetzung* beschreibt den überarbeiteten Prototypen und schildert die Integration in das Marking-Menü Framework. In der abschließenden Evaluation wird anhand des Anwenderfeedbacks bewertet, ob alle aufgestellten Anforderungen erfüllt wurden. Die anschließende Analyse bewertet den Schieberegler auf die Bedienbarkeit durch andere Eingabegeräte.

Kapitel 5.: *Zusammenfassung* wirft abschließend einen globalen Rückblick auf den Verlauf der Arbeit und gibt einen Ausblick auf zukünftige Arbeiten.

2 Stand der Technik

2.1 Pie-Menüs

Radial angeordnete Menüs wurden erstmals 1969 durch Wiseman, Lemke und Hiles [36] vorgestellt. In ihrem Artikel „PIXIE: A new approach to graphical man-machine communication“ beschrieben sie eine Menüstruktur, bei der um ein Auswahlkreuz herum eine Anzahl „lightbuttons“ kreisförmig angeordnet werden. Die beschriebenen „lightbuttons“ entsprechen auswählbarer Menüeinträge.

Das Menü enthält eine Ebene mit Einträgen, die das Menü in verschiedene Modi versetzt. Die angezeigten Menüeinträge sind abhängig von dem aktiven Modus des Menüs. Ein Menümodus kann als Definition eines Untermenüs gewertet werden, jeder Menümodus besitzt eine distinktive Anzahl von Menüeinträgen. Das Menü enthält nur solche Einträge, die derzeit für den aktiven Modus valide sind.

Die Auswahl eines Menüeintrags geschieht durch das exakte Treffen des Eintrags mit dem Eingabegerät. [36] Hopkins u. a. sehen im Gegensatz dazu vor, dass eine Auswahl in Pie-Menüs nicht auf Menüeinträgen selbst basieren, sondern über Winkel und Sektoren geschehen soll.

Die erste Definition eines Pie-Menüs geschah 1986 durch Hopkins u. a. in der E-Mail-Diskussion *Theta Menus Proposal and Pie Menu Designs*[17]. In dieser Diskussion erdachten sie eine Menüstruktur, bei der Menüeinträge eine initiale Äquidistanz zum Eingabegerät besitzen. Die Besonderheit des Menüs ist, dass eine Auswahl rein über die Richtung des Eintrags geschieht, in dem der Mauszeiger liegt, nachdem die aktive Maustaste gelöst wurde.

> The selection is indicated by the sector in which the cursor lies when the mouse button is released.
>
> — Hopkins u. a. *Theta Menus Proposal and Pie Menu Designs*, S. 1 [17]

Wie bei dem durch Wiseman, Lemke und Hiles [36] beschriebenen PIXIE Menü erscheinen Menüeinträge äquidistant um die Position des aktivierten Mauszeigers. Eine Auswahl geschieht, sobald der Mauszeiger in einem Sektor gelöst wird. Der aus Start- und Endposition berechnete Winkel wählt den Sektor aus, in dem sich der Menüeintrag befindet.

Die Besonderheit der Sektoren ist, dass diese einen Radius besitzen, der die gesamte Fläche des Bildschirms einnimmt. Eine Auswahl kann im Bereich des gesamten Sektors, also bis zum Rand des Bildschirms, erfolgen und setzt nicht das genaue Treffen eines Menüeintrags voraus.

Der Vorteil, eine Auswahl über Positionen und Sektoren zu treffen, ist die Umsetzung des Fitts'schen Gesetzes.

2.1.1 Fitts' Gesetz

Fitts' Gesetz ist ein 1954 durch Paul Fitts [9] beschriebenes Modell menschlicher Bewegungsabläufe. Nach diesem Gesetz ist die Zeit, die benötigt wird, ein Ziel zu erreichen, abhängig vom Verhältnis zwischen der Entfernung zum Ziel und der Breite des Ziels.

Bewegungsabläufe in Fitts' Gesetz werden als Übertragung von Informationen beschrieben. Jeder Bewegung wird ein Schwierigkeitsindex der Maßeinheit „bits" zugewiesen. Führt der menschliche Bewegungsapparat eine Bewegung aus, überträgt er in diesem Sinne eine Anzahl von „bits". Erfolgt zusätzlich die Betrachtung der Bewegungsdauer ergibt dies nach MacKenzie [25] die Maßeinheit „bits pro Sekunde".

Aus Fitts' Gesetz können drei Variablen abgeleitet werden. Der Schwierigkeitsindex ID gemessen in „bits", die Auslenkung (amplitude) der Bewegung A, sowie die Breite (width) des Ziels W. [24, S. 350]

Entsprechend kann Fitts' Gesetz in Formelschreibweise aufgestellt werden:

$$ID = log_2(2A/W)$$

Um eine Korrelation zwischen der Bewegungszeit MT sowie der Auslenkung und Zielbreite herstellen zu können, muss eine lineare Regressionsgleichung gebildet werden:

$$MT = a + b * log_2(2A/W)$$

Der Schnittpunkt mit der y-Achse entspricht a in der Maßeinheit „Sekunden", der Regressionskoeffizient b wird in „Sekunden pro bit" gemessen. [24, S. 353] Beide Konstanten sind abhängig von dem eingesetzten Eingabegerät. [23, S. 323]

Umgeformt ergibt Fitts' Gesetz:

$$MT = a + b * ID$$

Eine Weiterentwicklung des Fitts'schen Gesetzes erfolgte 1995 durch MacKenzie [26]. Die heute verbreiterte Berechnung des Schwierigkeitsindex lautet:

$$ID = log_2(A/W + 1)$$

In der weiterentwickelten Formel geht der Schwierigkeitsindex gegen 0, sobald A gegen 0 geht. Eine Auslenkung von 0 erreicht hingegen einen Schwierigkeitsindex von $-\infty$ in der 1954 durch Fitts aufgestellten Formel.

Da die Variablen A und W Distanzen entsprechen, enthält der Ausdruck des Logarithmus keine Maßeinheit. Nach Sasangohar, MacKenzie und Scott [31] wird diese aus der Wahl der Basis 2 des Logarithmus abgeleitet und entspricht der Maßeinheit „bits".

Fitts' Gesetz beschreibt demzufolge: Je größer der Schwierigkeitsindex einer Bewegung ist, umso höher fällt die Zeit aus, die zum Erreichen des Ziels benötigt wird.

2.1.2 Fitts' Gesetz in Pie-Menüs

Wie in Kapitel 2.1.: *Pie-Menüs* beschrieben reichen Sektoren der Menüeinträge bis zum Rand des Bildschirms. Eine Bewegung des Mauszeigers kann bis zu diesem, nicht darüber hinaus erfolgen, egal wie weit das Eingabegerät physikalisch bewegt wird. Im übertragenen Sinne ist ein Sektor eines Pie-Menüs in Bezug auf Fitts' Gesetz „unendlich" groß.

Unter Verwendung der MacKenzie-Formel [26] kann der Schwierigkeitsindex eines unendlich großen Sektors berechnet werden:

$$ID = log_2(A/\infty + 1) \Leftrightarrow log_2(1) = 0$$

Der Schwierigkeitsindex zum Treffen eines Menüeintrags beträgt somit theoretisch 0. Es ist zu beachten, dass ein unendlich großer Sektor real nicht umsetzbar ist. Trotzdem folgt aus den Eigenschaften eines Pie-Menüs ein geringerer Schwierigkeitsindex bei der Auswahl eines Eintrags im Vergleich zu einer linearen Menüstruktur.

Angewendet auf die Formel zur Berechnung der benötigten Zeit zum Erreichen einer Zielregion ist zu erkennen, dass diese bei einem unendlich großen Sektor rein von der Konstanten a abhängig ist.

$$MT = a + b * ID \Leftrightarrow a + b * 0 = a$$

Die Konstante a wird in *Sekunden* gemessen und entspricht laut MacKenzie [24, S. 353], je nach eingesetztem Eingabegerät des Experiments, der konstanten Dauer zur Betätigung des Eingabegeräts.

In Bezug auf Pie-Menüs bedeutet dies, dass die Dauer zum Erreichen eines Menüeintrags hauptsächlich von der Geschwindigkeit des Anwenders und des Eingabegeräts abhängig ist, und nicht durch das Menü beschränkt wird.

Fitts' Gesetz in Touch-Umgebunden Sasangohar, MacKenzie und Scott [31] evaluierten in ihrer Arbeit „Evaluation of Mouse and Touch Input for a Tabletop Display Using Fitts' Reciprocal Tapping Task" [31] die Performance zwischen Maus- und Touch-Eingaben in einer Tabletop-Umgebung.

Ergebnisse des Experiments zeigten, dass Touch-Eingaben mit $5.52 bit/s$ 40% performanter waren als Eingaben mit der Maus ($3.83 bit/s$). [31, S. 842] Touch-Eingaben waren konträr zu Mauseingaben fehleranfälliger, 9.8% gegenüber 2.1%. [31, S. 842] Fehler traten bei Touch-Eingaben vor allem bei dem Treffen kleiner Ziele auf.

An dieser Stelle kann die Vermutung angestellt werden, dass Touch-Interaktionen mit einem Pie-Menü performanter sein könnten als mit der Maus. Die Sektoren eines Pie-Menüs entsprechen die in der Studie zu treffenden Ziele. Aufgrund der Beschaffenheit von Pie-Menüs nehmen diese Sektoren vollflächig die Größe des Bildschirms ein. In Betrachtung des Ergebnisses der Studie von Sasangohar, MacKenzie und Scott [31] bedeutet dies, dass Fehlerraten vergleichbar mit der Maus sind, die Performance aber höher ausfallen müsste.

Eine formale Studie, die diese Annahme bestätigt existiert jedoch nicht.

2.1.3 Evaluation

Eine erste Evaluation der Performance zwischen Pie-Menüs im Vergleich zu linearen Menüs erfolgte 1988 durch Callahan u. a. in ihrem Beitrag „An empirical comparison of pie vs. linear menus“ [3].

Ergebnis Das Resultat der Versuchsstudie ergab, dass die Navigation durch Pie-Menüs, im Vergleich zu linearen Menüstrukturen, 15% schneller ausfiel und die Fehlerrate im Mittel um 40% sank. [3, S. 97]

Dieses Ergebnis wird zusätzlich durch Samp und Decker [30] bestätigt. In ihrer Studie fiel die Suchzeit in linearen Menüs schneller aus, die Auswahl von Einträgen aus radialen Menüs war im Gegensatz performanter.

Ein dem Pie-Menü zugrundeliegendes Problem ist die Begrenztheit in der Anzahl einsetzbarer Menüeinträge in einer Menüebene. [3, S. 100] [21, S. 486] [30, S. 3] Wo das lineare Menü in der Höhe des Bildschirms begrenzt ist, wird das Pie-Menü durch die Fehlerrate begrenzt. Zwar können hunderte Einträge in einem Pie-Menü angezeigt werden, eine performante und korrekte Auswahl ist in diesem Fall jedoch nicht mehr möglich.

Pie-Menüs mit 2, 4, 6, 8 oder 12 Einträgen werden in der Performance als am besten gewertet. [21, S. 483]

2.2 Pie-Menü Bedienarten

Nachfolgend soll eine Übersicht über den aktuellen Stand der Anwendungsgebiete für Pie-Menüs sowie der möglichen Interaktionsformen gegeben werden.

2.2.1 Mausbedienung

Insbesondere Computeranwendungen, in denen bestimmte Aktionen vermehrt ausgeführt werden, setzen die vorteilhaften Pie-Menüs um. Hierzu zählen vor allem der Bereich der Grafik- und CAD-Programme sowie Anwendungsstarter.

Blender Die bekannte 3D-Animations- und Modelingsoftware Blender führte die Verwendung von Pie-Menüs in Version 2.72 ein. [8]

Blender erlaubt die Erstellung und Anpassung eigener Pie-Menüs mit mehreren Menüebenen und Einträgen.

Abbildung 1 zeigt ein Pie-Menü der Anwendung Blender. Das Menü wird durch die Aktivierung einer konfigurierbaren Taste auf der Tastatur eingeblendet und steht anschließend in zwei Interaktionsmodi zur Verfügung. Bleibt die Maus stationär nach Betätigung der Aktivierungstaste, verweilt das Pie-Menü geöffnet, eine Auswahl im Menü erfolgt durch Bewegen und Aktivieren der Maustaste in dem Sektor eines Menüeintrags. Wird die Maus mit gedrückter Aktionstaste bewegt, erfolgt eine Auswahl des nächstgelegenen Menüelements, sobald die Aktionstaste gelöst wird. Nach einer getätigten Auswahl wird das Menü ausgeblendet.

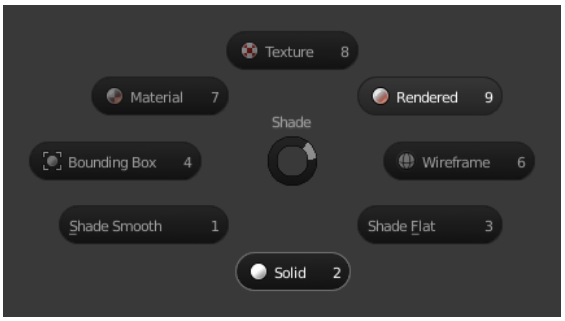


Abbildung 1: Pie-Menü für Shading-Operationen in der 3D-Modelingsoftware Blender. Quelle: *Blender 2.79 Manual* [2].

Menüeinträge können zusätzlich mit einer definierten Tastenkombination aus dem offenen Menü heraus ausgewählt werden. Die Tastenkombinationen sind in Abbildung 1 an den Zahlen 1 bis 9 (exklusive 5) zu erkennen. [2]

Um eine Auswahl zu treffen, reicht es aus, die Maus in die ungefähre Richtung eines Menüeintrags zu bewegen und zu aktivieren. Ein Eintrag muss nicht exakt durch die Maus getroffen werden, um ausgewählt werden zu können.

Gnome-Pie Gnome-Pie ist ein Anwendungsstarter für Linux mit frei konfigurierbaren Pie-Menüs. Neben dem Starten von Anwendungen können Menüs für die Steuerung von Medien oder der Simulation von Tastendrücken eingesetzt werden.

Ein Menü besteht aus mehreren definierbaren Menüeinträgen, diese werden bei der Aktivierung radial um den Mauszeiger herum eingeblendet. Einträge können zusätzlich per Tastenkombination aus dem aktiven Menü heraus ausgewählt werden. Das Pie-Menü bleibt nach der Aktivierung geöffnet, eine Auswahl erfolgt anschließend per Maus. [32]

Ein geöffnetes Gnome-Pie Pie-Menü wird in Abbildung 2 abgebildet. Erkennbar sind die acht definierten Menüeinträge. Der aktuell ausgewählte Eintrag erhält eine blaue Umrandung, zusätzlich wird der Name des Eintrags in der Mitte des Menüs angezeigt.

Für die Auswahl eines Menüeintrags muss nicht das Menüelement exakt getroffen werden, es reicht aus, das Eingabegerät in dem Sektor des Eintrags zu aktivieren.

Abbildung 2: Screenshot eines geöffneten Gnome-Pie Pie-Menüs. Quelle: *Gnome-Pie* [33].

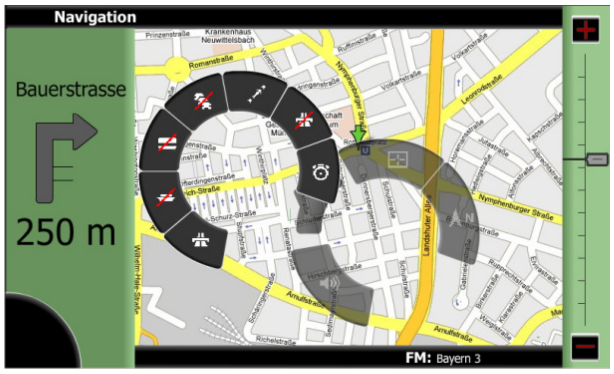


Abbildung 3: Pie-Menü im Navigationsbereich eines experimentellen BMW iDrive Systems. Quelle: „pieTouch“ [7, S. 5].

2.2.2 Touch

Die durch BMW veröffentlichten Studie „pieTouch“ untersuchte den Einsatz von Pie-Menüs im Rahmen eines In-Fahrzeuginformationssystems (IVIS). Die an das Pie-Menü gestellten Anforderungen sahen neben der Touchbedienung per „Point-and-Click“ zusätzlich die Gestenbedienung vor. Dies wird aus der Bedingung abgeleitet, die Konzentration des Fahrers während der Fahrt nicht auf die Bedienung des Menüs zu lenken. [7]

Das im BMW iDrive-System umgesetzte Pie-Menü erlaubt dem Fahrer während der Fahrt kontextabhängige Aktionen per Geste und Auswahl zu treffen. Abbildung 3 zeigt das Pie-Menü im Kontext des Navigationssystems. Zu erkennen ist die aus mehreren Ebenen bestehende Menüstruktur. Geöffnet ist ein Untermenü zur Änderung der Routing-Kriterien.

Im Gegensatz zu einem Pie-Menü am Desktop sind Einträge des Menüs nicht auf dem gesamten Kreisumfang abgebildet, sondern besitzen einen Bereich, in dem keine Menüeinträge angezeigt werden. Begründet ist dies aufgrund der Verdeckung des Bildschirms durch die Hand während der Anwendung.

Das kontextsensitive Menü wird durch einen Druck auf dem Touch-Bildschirm eingeblendet, die Auswahl eines Eintrags kann anschließend auf zwei Arten geschehen.

Per „Point-and-Click“ kann eine Auswahl getroffen werden, dies entspricht der Bedienung eines Pie-Menüs mit der Maus.

Zusätzlich erlaubt das Menü die Bedienung per Geste. In diesem Modus erfolgt eine Auswahl durch das Streichen in Richtung des gewünschten Menüeintrags. Wird die Geste gestoppt, bzw. der Finger gelöst, wird das entsprechende Untermenü an der Position des Fingers eingeblendet. Die Umsetzung dieses Modus bietet den Vorteil, dass eine Auswahl rein über die Richtung der Menüeinträge erfolgen kann und keine exakten Positionen getroffen werden müssen. Dies verringert die kognitive Belastung des Fahrers während der Bedienung des Menüs.

2.2.3 Weitere Anwendungen

Pie-Menüs in Virtual Reality Anwendungen werden hauptsächlich in wissenschaftlichen Beiträgen beschrieben. Ein Framework zur Umsetzung von Virtual Reality Anwendungen ist das durch die RWTH Aachen entwickelte „ViSTA Virtual Reality Toolkit“. [29]

Das ViSTA Toolkit stellt eine Bibliothek zur Erstellung von Pie-Menüs in Virtual Reality Anwendungen bereit. Programme, die auf ViSTA basieren, können die angebotenen Pie-Menüs umsetzen.

Der Bereich der Computerspiele setzt vermehrt auf die Vorteile der Bedienung durch Pie-Menüs. Die eingesetzten Menüs erlauben dem Spieler schnell auf Situationen zu reagieren und vereinfachen die Bedienung des Spiels, indem kontextsensitive Aktionen performant durchgeführt werden können.

Ein Beispiel ist das durch Capcom entwickelte Multiplayer Spiel Monster Hunter: World, das dem Spieler schnelle Reaktionen abverlangt. Aktionen des Spiels werden über verschiedene Pie-Menüs gesteuert, die Bedienung dieser erfolgt per Gamepad. [4]

2.2.4 Fazit

Wie aus den vorherigen Kapiteln ersichtlich wird, finden Pie-Menüs hauptsächlich am Computer per Maus eine Anwendung. Implementierungen in Touch- und VR-Umgebungen existieren, beschrieben werden diese aber hauptsächlich in wissenschaftlichen Beiträgen.

Pie-Menüs sind primär in solchen Anwendungsgebieten geeignet, in denen Aktionen schnell ausgeführt werden müssen oder bestimmte Befehle in hohem Maße angewendet werden. Die Schnelligkeit erlangt das Pie-Menü durch die Umsetzung des Fitts'schen Gesetzes.

Manche Umsetzungen erlauben die Interaktion per Touch-Eingabe. Dies entspricht im Allgemeinen einfacher Gesten, die als Mauseingaben gewertet werden. Berühren des Bildschirms entspricht einem Klick an der entsprechenden Stelle, wischen des Fingers über den Bildschirm entspricht der Bewegung des Mauszeigers.

2.3 Marking-Menüs

Die Idee des „mark ahead“ Modus eines Pie-Menüs wurde 1991 erstmalig durch Don Hopkins [16] beschrieben. Dieser Modus erlaubt die Auswahl eines Menüeintrags ohne, dass das Menü angezeigt wird. Kurtenbach und Buxton [20] führte diese Idee fort und entwickelten in seiner Dissertation „The limits of expert performance using hierarchic marking menus“ das Marking-Menü, welches eine Erweiterung der hierarchischen Pie-Menüs ist und den „mark ahead“-Modus konsequent umsetzt.

Im Gegensatz zu einem Pie-Menü, welches exklusiv per „Point-and-Click“ bedient wird, führt das Marking-Menü die Gestennavigation ein.

Kurtenbach und Buxton definieren das Marking-Menü als eine Auswahlmethode von Menüeinträgen eines Menüs. [20, S. 23] Die Bedienung erfolgt auf eine von zwei distinktiven Arten.

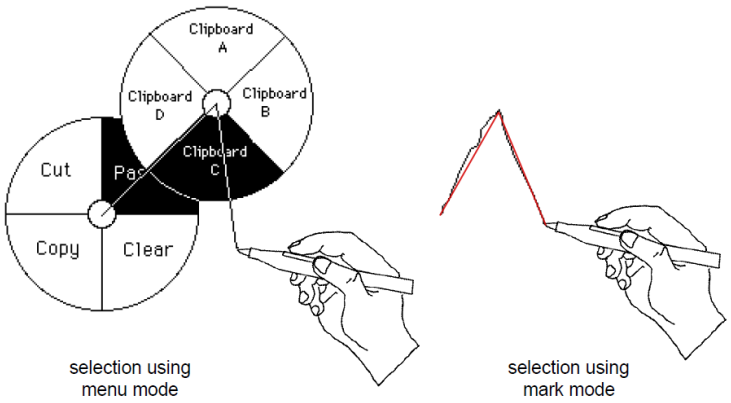


Abbildung 4: Die zwei Interaktionsmodi des Marking-Menüs nach Kurtenbach und Buxton. Der idealisierte Pfad ist in Rot dargestellt. Quelle: „The limits of expert performance using hierarchic marking menus“ [20, S. 24] (angepasst).

Der Modus „menu mode“ entspricht der klassischen Mausbedienung eines Pie-Menüs, in diesem erfolgt die Aktivierung eines Eintrags per Suche und Auswahl. Das Menü erscheint bei der Aktivierung um die Position des Eingabegeräts, das Eingabegerät wird anschließend in Richtung des gewünschten Eintrags bewegt. Eine grafische Hervorhebung kennzeichnet den aktiven Zustand des jeweiligen Menüeintrags.

Im Modus „mark mode“ erfolgt die Navigation per Gestensteuerung. Die Bewegungsrichtung einer Geste entspricht der Position eines Menüeintrags. In diesem Modus wird das Menü vom System nicht angezeigt, stattdessen wird der idealisierte Pfad des Eingabegeräts auf dem Monitor dargestellt. [34, S. 192]

Abbildung 4 visualisiert die zwei Interaktionsmodi nach der Definition Kurtenbach und Buxtons. Sowohl der Menümodus als auch der Marking-Modus führen zu der Auswahl des Menüeintrags „Clipboard C“. Während im ersten Modus das Menü grafisch dargestellt wird, ist im Marking-Modus einzig der zurückgelegte idealisierte Pfad des Eingabegeräts zu erkennen. Der idealisierte Pfad ist in Rot dargestellt. Während der Nutzung des Marking-Modus ist nur die idealisierte Form des Pfads sichtbar.

Marking-Menüs sind in ihrer Anwendung nicht auf eine Ebene beschränkt, sondern können wie das Pie-Menü eine Reihe von Hierarchien enthalten. Untermenüs werden während des Menü-Modus an der Position des Eingabegeräts zum Zeitpunkt der Auswahl geöffnet. Um einen Menüeintrag eines Untermenüs im Marking-Modus auszuwählen, muss eine „Zick-Zack“-Geste gezeichnet werden, die den Winkeln der Menüeinträge entspricht. Die Gesten eines Marking-Menüs bestehen aus mehreren zusammengesetzten Liniensegmenten unterschiedlicher Winkel.

Nach der Definition von Kurtenbach und Buxton [20, S. 25] können Menüeintrag einer Geste verifiziert werden. Dazu verweilt das Eingabegerät am Ende einer Geste im aktiven Zustand, nach einer bestimmten Dauer wird das Menü entlang der Geste eingeblendet. Das angezeigte Menü entspricht anschließend der Darstellung des Menüs im Menü-Modus.

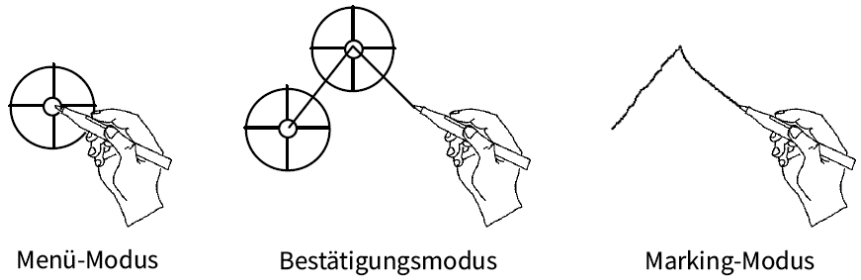


Abbildung 5: Übergang eines Anwenders vom Anfänger zum Experten. Quelle: „The limits of expert performance using hierarchic marking menus“ [20, S. 38].

2.3.1 Stärken

Die Stärke des Marking-Menüs ist der kontinuierliche Übergang des Anwenders vom Anfänger zum Experten.

Nach Kurtenbach und Buxton [20] durchläuft ein Anwender bei der Benutzung eines Marking-Menüs drei Phasen, vom Anfänger zum Experten. Diese sind in Abbildung 5 abgebildet. Anfänger starten bei der Benutzung des Marking-Menüs im Menü-Modus. Dem Anfänger sind weder die Positionen der Menüeinträge noch die Navigation per Geste bekannt. Einträge des Menüs werden per Suchen und Auswählen ausgewählt.

In der intermediären Phase arbeitet der Anwender mit Gesten, ist aber teilweise noch unsicher, ob eine Geste zu der gewünschten Auswahl führt.

Der Übergang zur Expertennutzung entspricht der dritten und letzten Phase. Der Anwender arbeitet fast ausschließlich im Marking-Modus und benötigt kein visuelles Feedback mehr.

Die Anwendung des Marking-Modus verstärkt die Assoziation von Geste mit Menüeintrag. Dies geschieht vor allem durch den Übergang der Bewegung zum Erreichen eines Menüeintrags in das Muskelgedächtnis. Daraus folgt ein stetiger Anstieg der Auswahlgeschwindigkeit sowie eine abfallende Fehlerrate. [20, S. 76]

2.3.2 Schwächen

Die Performance eines Marking-Menüs korreliert mit der Anzahl der Einträge eines Menüs. Je mehr Einträge in einem Menü enthalten sind, umso höher fallen die Auswahlzeiten und Fehlerraten aus. [20, S. 70] Dies wird zusätzlich durch Kurtenbach [19] bestätigt. Die Auswahlzeit eines Menüs mit vier Einträgen lag im Mittel bei $0.8 Sekunden$, zwölf Einträge führten zu einer Auswahlzeit von $1.2 Sekunden$, die Anzahl der falsch ausgewählten Einträge stieg von 0 auf 2. [19, S. 14] Ergebnisse der Studie von Kurtenbach und Buxton ergaben, dass ein Marking-Menü performanter bei vier, acht oder zwölf Menüeinträgen ist. [20, S. 81]

2.4 User Interface Widgets

Die durch Kurtenbach aufgestellte Definition eines Marking-Menüs sieht im Grunde ausschließlich die Umsetzung eines Widgets zur Befehlsausführung vor.

User Interfaces werden im Regelfall aus einigen standardisierten Steuerelementen gebildet. Das folgende Kapitel gibt eine kompakte Übersicht dieser Widgets und analysiert die in Pie-Menüs implementierten Steuerelemente. Eine anschließende Bewertung klassifiziert die Widgets entsprechend ihrer Trivialität der Umsetzbarkeit.

2.4.1 Standard User Interface Widgets

Viele Betriebssysteme und Interface Frameworks setzen eine Liste standardisierter Widgets um. Garrett [12, S. 116ff] fasst diese Grundelemente wie folgt zusammen:

Checkbox Die Checkbox ist ein Kontrollelement, welches einem booleschen Wert entspricht. Dieses Widget kann einen von zwei Werten annehmen. Der Zustand „ausgewählt“ entspricht dem Wert „Wahr“, im nicht ausgewählten Zustand entspricht die Checkbox dem Wert „Falsch“. Mehrere Checkboxen sind untereinander nicht abhängig.

Radio Button Das Widget des Radio Buttons entspricht wie dem Element der Checkbox der Repräsentation eines booleschen Wertes. Im Gegensatz zu der Checkbox, erfordert der Radio Button den gruppierten Einsatz von mindestens zwei Elementen. Innerhalb einer Gruppe sind die Widgets untereinander abhängig, zu einem Zeitpunkt kann nur ein Radio Button aktiv sein.

Textfeld Das Widget des Textfelds erlaubt es arbiträre alphanumerische Werte einzugeben. Je nach Anwendungsfall kann der Funktionsumfang auf alleinig alphabetische oder numerische Werte beschränkt werden.

Dropdown-Liste Die Dropdown-Liste vereint eine bestimmte Anzahl von Radio Button in einer kompakten Form. Werte werden in Listenform angezeigt, eine Auswahl ist auf ein aktives Element beschränkt.

List Box Ähnlich der Dropdown-Liste beinhaltet die List Box eine Reihe von Checkboxen. Bei diesem Widget werden annehmbare Werte in einer Liste angezeigt, eine Auswahl kann aus einem oder mehrerer Elemente dieser Liste getroffen werden.

Aktionsknopf Der Aktionsknopf führt bei Aktivierung eine vordefinierte Aktion aus.

Schieberegler Das Widget des Schiebereglers wird im Allgemeinen als horizontaler oder vertikaler Form dargestellt. Der Funktionsumfang umfasst die Anpassung und das Setzen eines Werts. Schieberegler können in mehreren Variationen und Funktionen auftreten. Am häufigsten werden Schieberegler zur Eingabe numerischer Werte eingesetzt.

Widget	Trivial	Umsetzung
Checkbox	✓	Kulik u. a. [18]
Radio Button	✓	Kulik u. a. [18]
Textfeld	✗	Guimbretiére und Winograd [14]
Dropdown-Liste	✓	-
List Box	✓	-
Aktionsbutton	✓	-
Schieberegler	✗	Kulik u. a. [18]

Tabelle 1: Übersicht Interface Widgets in Pie-Menüs. Trivial implementierbare Elemente sind mit einem (✓) versehen. Nichttrivial umsetzbare Widgets sind mit (✗) gekennzeichnet. Die Spalte „Umsetzung“ enthält Referenzen zu Beiträgen mit entsprechenden Implementierungen der Widgets.

2.4.2 Widgets in Pie-Menüs

Die in Kapitel 2.4.: *User Interface Widgets* beschriebenen Widgets sollen unter dem Aspekt der Umsetzbarkeit in Pie-Menüs bewertet werden.

Checkbox Eine Checkbox entspricht im Kontext eines Pie-Menüs einem Menüeintrag mit veränderbarem Zustand. Die Erweiterung des Funktionsumfangs eines Menüeintrags entspricht der Implementierung dieses Widgets. Eine Umsetzung ist in Folge trivial und wird durch Kulik u. a. [18] beschrieben.

Radio Button Der Radio Button ist wie die Checkbox ein Menüeintrag mit veränderbarem Zustand. Die Anforderungen des Widgets setzen den Einsatz von mindestens zwei Radio Buttons voraus. Dieses Widget ist in der Implementierung trivial, benötigte Funktionalitäten können im Rahmen eines Menüeintrags umgesetzt werden. Kulik u. a. [18] beschreiben eine Form der Umsetzung.

Textfeld Das Textfeld kann in einem Pie-Menü nicht trivial umgesetzt werden, für dieses Widget müssen für eine performante Integration neue Menüstrukturen geschaffen werden. Eine Umsetzung auf Basis normaler Menüeinträge beschreiben Guimbretiére und Winograd [14].

Dropdown-Liste Die Dropdown-Liste kann in Form eines Untermenüs umgesetzt werden. Die Anzahl der Listeneinträge entspricht der Anzahl der Einträge des Untermenüs.

Die Implementierung ist trivial, sofern die Menge der Einträge eine bestimmte Anzahl nicht überschreitet. Bei einer Überschreitung dieser Anzahl muss unter Umständen eine Segmentierung in zusätzliche Untermenüs stattfinden.

List Box Erkenntnisse der Dropdown-Liste können auf das Widget der List Box angewendet werden. Ein Untermenü entspricht den Einträgen der Liste, eine Segmentierung in zusätzliche Menüs muss stattfinden, sobald die Anzahl der Einträge eine performante und fehlerfreie Auswahl nicht mehr erlaubt.

Aktionsknopf Der Aktionsknopf entspricht der Definition von Menüeinträgen eines Pie-Menüs. Eine Auswahl eines Eintrags ist die Aktivierung eines Aktionsbutton.

Schieberegler Das Widget des Schiebereglers ist gleichermaßen wie das Textfeld nicht trivial in einem Pie-Menü umsetzbar. Für dieses Widget müssen neue Menüstrukturen geschaffen werden. Kulik u. a. [18, S. 651] beschreiben die simple Implementierung eines Schiebereglers, der in Tests als zu inakkurat bewertet wurde.

Übersicht Zusammengefasst ergibt Tabelle 1 den Stand der Technik über die Trivialität und Umsetzbarkeit von Steuerelementen in Pie-Menüs. Bis auf die Widgets Textfeld und Schieberegler können alle Steuerelemente trivial in der Menüstruktur des Pie- bzw. Marking-Menüs umgesetzt werden.

2.5 Schieberegler

Nach Galitz [11] entspricht der Schieberegler einer Skala, die den Betrag oder Grad einer Größe, oder die Qualität auf einem Kontinuum anzeigt. [11, S. 518]

Ein Schieberegler besteht mindestens aus zwei Elementen, der Achse und dem Indikator. Die Achse entspricht dem Bereich aller annehmbarer Werte, sie ist konventionell horizontal oder vertikal angeordnet. Der Indikator läuft entlang der Achse und visualisiert die relative Position auf dem Wertebereich.

Die Auswahl eines Werts geschieht über die Bewegung des Indikators auf der Achse. Eine Veränderung der Position kann zusätzlich durch einen Klick auf der Achse geschehen, der Indikator springt in diesem Fall auf die ausgewählte Position.

Die Auswahl eines Wertes durch einen Schieberegler ist nach Vora [35, S. 245] fehleranfällig. Begründet wird dies durch die fehlende Erkennbarkeit valider Werte, der Präzision und Schrittgröße des Schiebereglers vor der ersten Anwendung.

Der Fehleranfälligkeit kann durch Umsetzung einiger Funktionalitäten in einem Schieberegler entgegengewirkt werden. Eine schrittweise Veränderung des Indikators durch dedizierte Knöpfe Galitz [11, S. 520] oder Tasten am Eingabegerät Vora [35, S. 246] erlauben eine exakte Einstellung des Schiebereglerwerts.

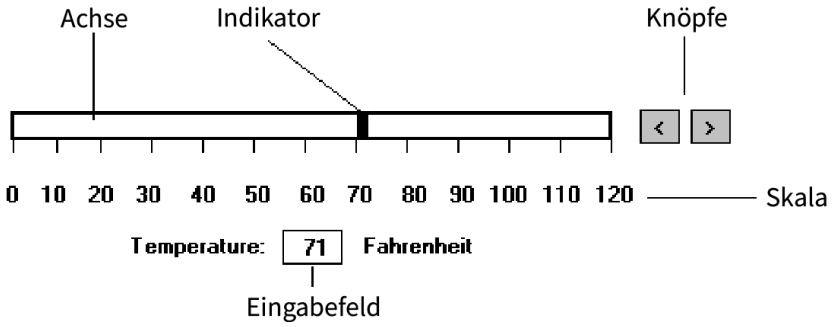


Abbildung 6: Elemente eines Schiebereglers. Quelle: *The essential guide to user interface design* [11, S. 519] (angepasst).

Abbildung 6 visualisiert die Elemente eines Schiebereglers. Zu erkennen sind die Achse mit einem Wertebereich von 0 bis 120, der Indikator mit Werteposition 71, zwei Knöpfen zur schrittweisen Veränderung des Werts sowie einem Eingabefeld zur direkten Eingabe des Werts.

Das Element des Schiebereglers findet seine beste Anwendung, wenn der exakte Eingabewert nicht relevant ist, sondern ein ungefährer Wert ausreicht. [15]

2.5.1 Formen

Nichtlinear Ist die Änderungsrate des Werts eines Schiebereglers nicht konstant, sondern analog einer mathematischen Funktion, entspricht dies einem nicht linearen Schieberegler.

Diskret Werte eines diskreten Schiebereglers können nur in bestimmten Schritten eingestellt werden. Im Allgemeinen sind alle Schieberegler diskret, da ständig eine systemseitige Schrittgröße definiert ist.

Die Schrittgröße muss keiner linearen Änderungsrate $\{0, 2, 4, 6, 8\}$ entsprechen, sondern kann durch eine mathematische Funktion wie 2^n beschrieben werden.

Ordinal Neben dem Schieberegler zur numerischen Werteeingabe existieren noch zusätzliche Formen, darunter die Form zur ordinalen Werteauswahl.

Ein Beispiel ist der von Ahlberg und Shneiderman [1] entwickelte „Alphaslider“. Bei dieser Form des Schiebereglers entspricht der Wertebereich dem Alphabet von $A - Z$. Die Entwicklung des Schiebereglers fand statt, um schnell eine Auswahl aus einer größeren Liste alphanumerischer Einträge treffen zu können.

Die Liste wird analog der Position des Indikators gefiltert, der Indikatorwert entspricht dem ersten Zeichen, der in der Liste angezeigten Einträge.

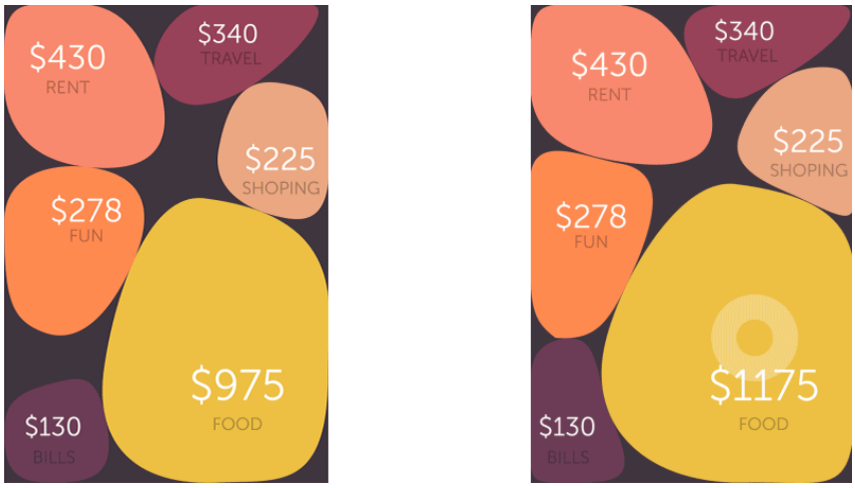


Abbildung 7: Schieberegler in Form mehrerer disproportionaler Figuren. Quelle: *Expense Input Interaction* [5].

2.5.2 Weitergeführte Formen

Die Funktionalität des Schiebereglers ist nicht auf die Elemente Achse, Indikator und Wertebereich begrenzt. Das Kernelement eines Schiebereglers, das Verändern eines Werts, kann aufgeschlüsselt und in einem anderen Kontext angewendet werden.

Im Kontext einer Anwendung zum Nachverfolgen monatlicher Ausgaben kann ein Schieberegler wie in Abbildung 7 genutzt werden. Der Schieberegler besteht in dieser App nicht aus Achse mit Indikator, sondern wird als disproportionale Figur dargestellt.

Mehrere Schieberegler sind in der App visualisiert. Der Wert des Schiebereglers wird analog der Bewegungsrichtung der ausgeführten Geste geändert. Ein Wischen nach oben erhöht den Wert, ein Wischen nach unten verringert den Wert.

Die exakte Wertigkeit eines Reglers wird numerisch in der Figur angezeigt, zusätzlich wird eine ungefähre Wertigkeit über die Größe der Figur kommuniziert. Während in der ersten Ansicht die Figur „Food“ noch verhältnismäßig viel Freiraum zu den anderen Elementen besitzt, ist der Freiraum in der zweiten Ansicht geringer und die Wertigkeit höher.

2.5.3 Integration in Pie-Menüs

Die Integration eines Schiebereglers in ein Pie-Menü ist nicht trivial. Interaktionen mit einem Schieberegler bestehen nicht aus einzelnen Kommandos wie bei einem Aktionsknopf, sondern erfordern verschiedene dedizierte Eingaben.

FaST Sliders Bei dem durch McGuffin, Burtnyk und Kurtenbach [27] entwickelten FaST Slider wurde ein vertikaler Schieberegler mit einem Marking-Menü kombiniert.

Das Akronym FaST steht für „Flick and Slide or Tweak“ [27, S. 2], „Schnipsen“ und „Gleiten“ oder „Anpassen“.

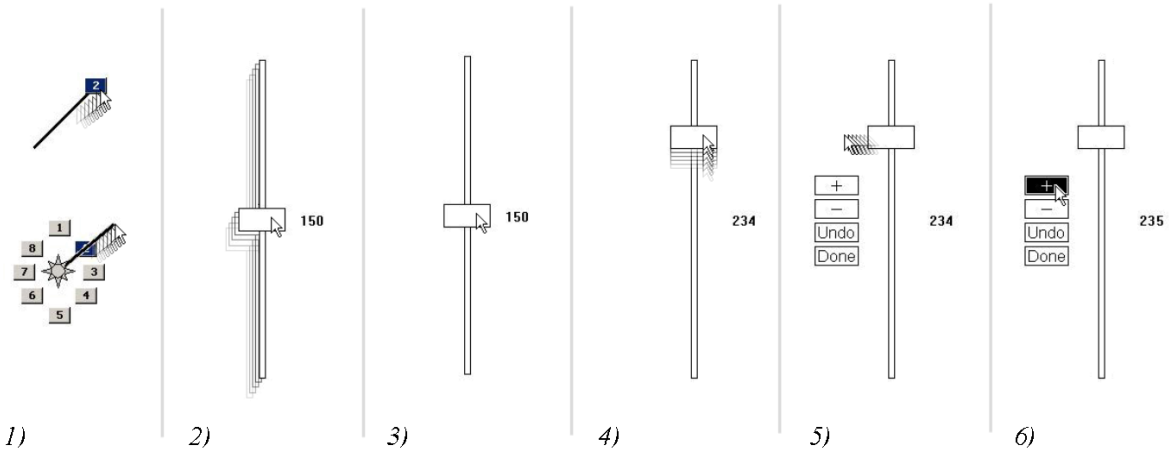


Abbildung 8: Interaktionsablauf mit dem FaST Slider. Quelle: „FaST Sliders: Integrating Marking Menus and the Adjustment of Continuous Values“ [27, S. 2].

Dies beschreibt die drei eigenständigen Interaktionsschritte bei der Benutzung des Schiebereglers.

Der Erste Schritt des „Schnipsen“ entspricht der Auswahl des Schiebereglers aus dem Marking-Menü. Im weiteren Schritt des „Gleitens“ wird der eigentliche Wert des Reglers angepasst. Der optionale Schritt „Anpassen“ erlaubt eine Feinjustierung des Werts.

Der Interaktionsablauf des FaST Sliders wird in Abbildung 8 grafisch dargestellt.

Zunächst wird eine Auswahl per Geste in einem Marking-Menü getroffen (1). Existiert hinter einem Menüeintrag ein Schieberegler, wird das gesamte Menü ausgeblendet und an der Position des Eingabegeräts erscheint ein Schieberegler, der der Bewegung der Maus folgt (2). Nach Erreichen der gewünschten Schiebereglerposition muss das Eingabegerät erneut aktiviert werden. Der Schieberegler verweilt anschließend an der gewählten Position (3). Mit der Maus kann der Wert des Schiebereglers über den Indikator verändert werden (4).

Das Lösen der Maustaste blendet den Schieberegler aus und die Interaktion wird beendet.

Eine lotrechte Bewegung der Maus zu dem Schieberegler blendet Kontrollelemente zur Feinjustierung des Werts ein (6). Die Bedienung der Kontrollelemente erfolgt per Klick, die Interaktion mit dem Schieberegler wird durch einen dedizierten „Done“-Knopf beendet.

McGuffin, Burtnyk und Kurtenbach beschreiben das System des FaST Sliders als reine Kombination eines Schiebereglers und Marking-Menüs. [27, S. 2ff] Die Vorteile eines Marking-Menüs werden während der Interaktion mit dem Schieberegler nicht umgesetzt.

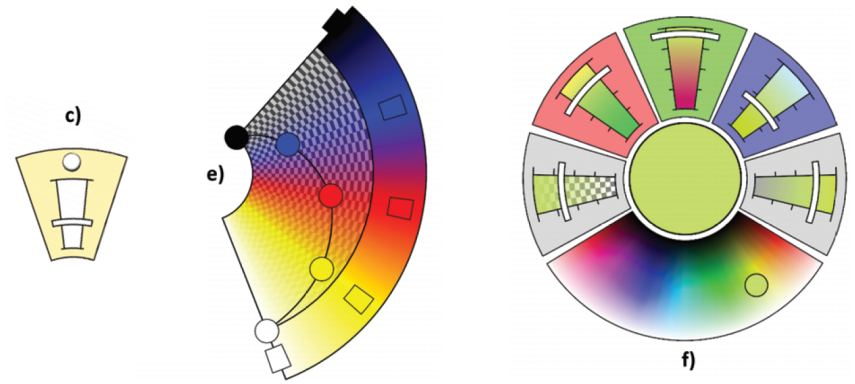


Abbildung 9: Schieberegler des ViSTA VR Toolkit. Quelle: „Extended pie menus for immersive virtual environments“ [13, S. 649] (Ausschnitt).

ViSTA-Toolkit Das von der RWTH Aachen entwickelte VR-Toolkit ViSTA erlaubt die Umsetzung von Schiebereglern in Virtual Reality Pie-Menüs. Entwickelt wurden diese im Rahmen der Arbeit „Extended pie menus for immersive virtual environments“. [13]

Die Bedienung der Menüs ist beschränkt auf den Einsatz von VR-Controllern. Aufgrund der eingesetzten Interaktion per „Pick Ray“, kann die Nutzung mit der Maus dennoch abgeleitet werden. [13, S. 649]

Bei der Bedienung per „Pick Ray“ werden Menüeinträge durch einen grafisch dargestellten Strahl, der vom VR-Controller ausgeht, ausgewählt. Der Strahl entspringt dem Zentrum des Controllers, der Schnittpunkt zwischen dem Strahl und dem Pie-Menü determiniert das ausgewählte Menüelement. [13, S. 646]

Die im ViSTA-Toolkit umgesetzten Schieberegler sind in Abbildung 9 abgebildet.

Der Standardschieberegler kann in mehreren Funktionsformen vorliegen. Eine Werteauswahl zwischen einem Minimal- und Maximalwert ist in (c) dargestellt. Ein Menü mit Schiebereglern zur Farbänderung ist sowohl und (e) als auch in (f) abgebildet.

Der Schieberegler in (e) liegt in der gesonderten Form der „color map“ vor. Jedes Rechteck kann frei auf dem Farbspektrum bewegt werden, die zugehörigen beweglichen Punkte stellen die Deckkraft der Farbe ein. [13, S. 650]

Das in (f) dargestellte Menü enthält fünf Schieberegler zur Farbänderung sowie ein frei beweglicher Schieberegler in einem Farbspektrum.

Alle im ViSTA-Toolkit umgesetzten Schieberegler erlauben eine ungefähre Einstellung des hinterlegten Werts. Funktionen zur Feinjustierung oder die Anzeige des genauen Werts wurden in der Umsetzung nicht betrachtet.

Anwender bemängelten in Tests die fehlende Visualisierung des geänderten Werts und beschrieben die Interaktion der Schieberegler teilweise als zu ungenau. Ein anderer Kritikpunkt war die fehlende Visualisierung des Wertebereichs. [13, S. 651]

2.6 Zusammenfassung

Aus der bisherigen Darstellung kann das Fazit gezogen werden, dass Pie-Menüs gegenüber linearer Menüs Vorteile in der performanteren Bedienbarkeit und geringer ausfallenden Fehleranfälligkeit bieten. Dies wird vor allem durch die exakte Umsetzung des Fitts'schen Gesetz begründet.

Trotz erwiesener Vorteile ist der Einsatz von Pie-Menüs hauptsächlich auf die Anwendungsgebiete der CAD-Programme und Computerspiele beschränkt. Der Performancevorteil von Pie-Menüs ist ersichtlich bei der häufigen oder schnellen Ausführung von Aktionen.

Die auf der Idee des Pie-Menüs aufbauenden Marking-Menüs erweitern diese um die Bedienung per Geste und bieten den Vorteil des selbstverstärkenden Lerneffekts. Je öfter ein Marking-Menü genutzt wird, umso schneller fallen anschließende Auswahlen aus. Die Bewegung zum Erreichen eines Menüeintrags geht stetig in das Muskelgedächtnis über, eine Auswahl kann blind erfolgen.

Viele vom Betriebssystem bereitgestellte Steuerelemente finden in verschiedenen User Interfaces eine Anwendung. Ein Großteil dieser kann trivial in Pie- bzw. Marking-Menüs umgesetzt werden. Nichttrivial im Kontrast sind Widgets, die dedizierte Eingaben während der Anwendung fordern. Die in den verwandten Arbeiten beschriebenen Umsetzungen von Schiebereglern in Marking-Menüs sind nicht zufriedenstellend, in ihnen fehlen Funktionalitäten oder sie erlauben keine ausreichende Performance und Präzision in der Bedienung.

Das Element des Schiebereglers soll deshalb in dieser Arbeit beschrieben, analysiert und in einem Marking-Menü umgesetzt werden.

3 Marking-Menü Framework

Aufgrund der mangelnden Verfügbarkeit an Marking-Menü Frameworks wurde vor dieser Arbeit ein entsprechendes Framework entwickelt. Das folgende Kapitel beschreibt den Funktionsumfang und die Struktur der entwickelten Lösung.

Das Design und die Evaluation des Frameworks sind nicht Teil dieser Arbeit.

Vorbereitend zu der Entwicklung des Marking-Menü Frameworks fand eine Untersuchung auf bestehende Umsetzungen statt. Das Ergebnis der eingehenden Recherche ergab, dass keine Lösungen für Marking-Menüs existieren, die die Nutzung beliebig vieler Menüebenen erlauben, und darüber hinaus die Gesten- und Klicknavigation unterstützen.

Damit die Fragestellung dieser Arbeit beantwortet werden kann, musste als Grundlage ein entsprechendes Marking-Menü Framework geschaffen werden.

3.1 Grundlagen

Mehrere Anforderungen wurden eingangs an das zu entwickelnde Marking-Menü Framework gestellt. Eine Aufteilung der Anforderungen fand in die Kategorien funktional und nichtfunktional statt.

3.1.1 Funktionale Anforderungen

Die funktionalen Anforderungen entsprechen dem realen Funktionsumfang des Marking-Menüs. Funktional soll das Marking-Menü alle Anforderungen abdecken, wie sie durch Kurtenbach und Buxton [20] definiert wurden. Die initiale Definition Kurtenbachs wurde um fünf funktionale und drei nichtfunktionale Anforderungen erweitert.

Einsatz in verschiedenen Umgebungen Das Framework soll in verschiedene Umgebungen integrierbar sein. Die Nutzung auf einer Webseite soll genauso realisierbar sein wie der Einsatz in einer Desktopumgebung.

Simple Menüstruktur Die Struktur eines Menüs soll einfach definiert werden können. Die Definition eines Menüeintrags soll einen beschreibenden Text und Icon umfassen und zusätzlich beliebige viele Untermenüs enthalten können.

Unterstützung verschiedener Eingabegeräte Neben dem Einsatz in verschiedenen Umgebungen soll das Menü nicht auf die Bedienung durch ein Eingabegerät begrenzt sein.

Verwendung von Icons Damit ein Anwender möglichst schnell erkennen kann welche Funktionen oder Einträge hinter einem Menüeintrag liegen, soll für jeden Eintrag ein Icon definiert werden können.

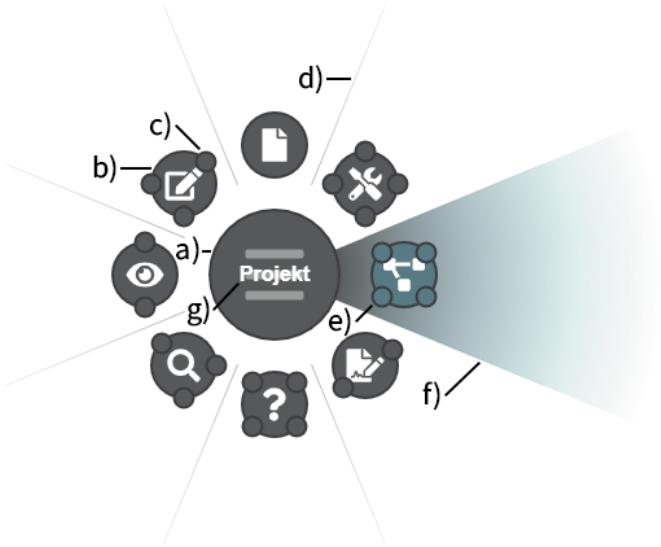


Abbildung 10: Grafische Elemente eines Menüeintrags. Das Hauptelement des Menüs ist unter (a) zu sehen und enthält acht Kind-Elemente (b). Zusätzlich sind die Kind-Elemente der zweiten Ebene visualisiert (c). Die Sektoren der Menüeinträge werden durch Trennlinien (d) unterteilt. Das aktuell ausgewählte Element wird andersfarbig hervorgehoben (e) und färbt seinen Sektor mit einem Farbverlauf (f) ein. Jeder Menüeintrag enthält ein Icon, dieses wird mittig im jeweiligen Menüelement angezeigt. Um eine visuelle Überladung zu vermeiden werden Icons in der zweiten Ebene (c) nicht mehr angezeigt. Der beschreibenden Texte des ausgewählten Menüeintrags (e) wird im Elternelement eingeblendet (g). Quelle: *Eigener Screenshot.*

Zustandskommunikation Das Menü soll Informationen zum aktuellen Zustand mittels einer definierten Schnittstelle kommunizieren. Die Informationen sollen Daten über ausgewählte Menüeinträge, Navigationsentscheidungen oder über das Schließen des Menüs enthalten.

3.1.2 Nichtfunktionale Anforderungen

Die nichtfunktionalen Anforderungen gehen über die funktionalen Anforderungen hinaus und beschreiben die Leistung des Menüs.

Schnelle Übersicht In einem Menü soll schnell erkennbar sein, welche Einträge vorhanden sind und ob diese zu einem Befehl oder Untermenü führen.

Bedienbarkeit Die Bedienung des Menüs soll intuitiv sein.

Dynamisches „Look and Feel“ Die Bedienung des Menüs soll dynamisch ablaufen, das Menü soll auf getätigte Eingaben „reagieren“ und nicht statisch wirken.

3.2 Umsetzung

Um ein möglichst breites Einsatzgebiet abdecken zu können, wurde als Grundlage für die Entwicklung eine Umsetzung mit aktuellen Webtechniken gewählt. Der Einsatz aktueller Frameworks wie Electron oder Gnome-Shell erlaubt vermehrt die Nutzung von Webtechniken für die Erstellung von Benutzeroberflächen in Desktop-Umgebungen. Der Einsatz in Webumgebungen wird nativ über den Browser abgedeckt. Die Programmierung erfolgt in der Sprache TypeScript, einer JavaScript-Implementierung von Microsoft die auf eine starke Typisierung setzt, sowie der HTML Canvas zur Visualisierung des Menüs. Abbildung 10 zeigt die Darstellung des entwickelten Menüs.

Das Framework wurde in zwei verschiedenen Strukturen aufgeteilt. Die klassenlogische Struktur definiert die Abhängigkeiten und Vererbungen der Klassen untereinander. Die grafisch-logische Strukturierung beschreibt die Organisation und Abhängigkeiten der angezeigten Elemente des Menüs.

3.2.1 Klassenstruktur

Die Struktur des Menüs besteht primär aus zwei Hauptklassen. Die Klasse „Menu“ enthält die Menüstruktur in Form mehrerer „MenuItem“ und verarbeitet die Daten des Eingabegeräts.

Die Klasse „MenuItem“ entspricht einem Menüeintrag mit Icon und beschreibendem Text, sie kann 0 bis n zusätzliche Menüeinträge als Kind-Elemente enthalten. Enthält ein Menüeintrag keine zusätzlichen Elemente, wird dieser Eintrag als ausführbarer Befehl gewertet.

3.2.2 Menüstruktur

Die Menüstruktur wird durch ein JSON-Objekt repräsentiert. Dies hat den Vorteil die Strukturdefinition simpel zu halten und zusätzlich eine programmatische Generation ermöglichen zu können. Jeder Menüeintrag enthält eine eindeutig identifizierende Bezeichnung, einen beschreibenden Text sowie ein Icon.

Die Position eines Menüeintrags wird über die Angabe eines Winkels gesteuert. Das Winkelsystem des Frameworks definiert 360° rechtsdrehend mit 0° als Norden und 180° als Süden. Dieses System wurde gewählt, um die Anordnung der Menüeinträge für den Anwender zu vereinfachen. Die Abbildung des Winkelsystems auf die Himmelsrichtungen ist intuitiv und benötigt für den Anwender keine zusätzliche Beschreibung.

Die Wahl den Winkel eines Menüeintrags explizit anzugeben erfolgte, um die Möglichkeit offenzulassen, eine bestehende Menüstruktur zu einem späteren Zeitpunkt ändern zu können, ohne dass Einträge in einem Menü verschoben werden. Dieselben Einträge bleiben nach einer Änderung weiter über dieselben Gesten im Menü zu erreichen.

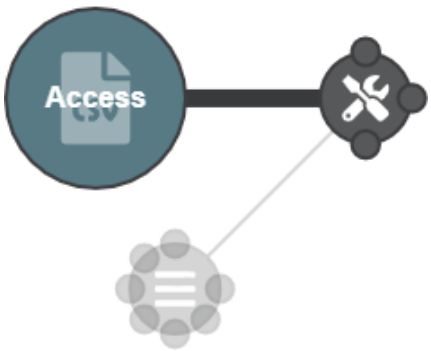


Abbildung 11: Selektion eines Eintrags im Marking-Menü. Der ausgewählte Eintrag ist bläulich hervorgehoben. Zu erkennen ist, dass alle nicht auswählbaren Menüeinträge, transparent ausgegraut sind. Im Falle einer Falschauswahl kann während der Bedienung per Geste eine Rücknavigation in das Elternelement mit aktiv gehaltenem Eingabegerät erfolgen. Quelle: *Eigener Screenshot.*

3.2.3 Szenengraph

Neben der klassenlogischen Organisation des Menüs erfolgt zusätzlich eine grafisch-logische Gruppierung der angezeigten Elemente des Menüs. Diese werden durch einen Szenengraph beschrieben.

Ein Szenengraph ist eine Datenstruktur, die die räumliche und logische Anordnung sowie die Abhängigkeit zwischen Elementen abbildet.

Der durch das Paper.js [22] Framework bereitgestellte Szenegraph kann mit Grafikanwendungen wie Adobe PhotoShop oder Illustrator verglichen werden.

In einer Szene können mehrere Ebenen existieren, jede Ebene kann aus mehreren Pfaden und Gruppen bestehen. Gruppen können als einziges Konstrukt zusätzliche Kind-Elemente enthalten und wenden jegliche Transformationen automatisch auf alle enthaltenen Elemente an.

3.2.4 Interaktion

Das Framework erlaubt sowohl die Interaktion per Klick als auch die Bedienung per Geste. Zusätzlich erfolgte die Integration einer „Mischnavigation“.

Die von Kurtenbach und Buxton aufgestellte Definition des Marking-Menüs sieht exklusiv eine aktive Art der Bedienung vor. Eine Auswahl geschieht entweder per Geste oder Klick nicht aber per Gesten- und Klicknavigation.

In der neu eingeführten Mischnavigation wird die Möglichkeit geschaffen während einer Auswahl fließend zwischen den zwei Bedienmodi zu wechseln. Eine Navigation in ein Untermenü kann per Geste erfolgen, ab dort kann auf die Klicknavigation gewechselt werden. Der Wechsel des Bedienungsmodus ist in jedem aktiven Menü mit Kind-Elementen möglich.

Die Navigation in ein Untermenü fügt dem Menü einen virtuellen Menüeintrag in Richtung des Elternelements hinzu. Dieser Eintrag besitzt kein dargestelltes Menüelement, sondern einzig einen wie in Abbildung 10 unter (f) dargestellten Farbverlauf, der bei Aktivierung oder Bewegung des Eingabegeräts über diesen eingeblendet wird. Der Menüeintrag dient der Navigation zurück zum Elternelement. Wird das Eingabegerät in dem Sektor des Eintrags aktiviert, erfolgt die Navigation zurück, das Elternelement verschiebt sich auf die Position der Aktivierung des Eingabegeräts. Abbildung 11 zeigt die Selektion eines Menüeintrags.

Die Implementierung des Frameworks erfolgte unter der Beachtung, verschiedene Eingabegeräte unterstützen zu können. Über eine Schnittstelle können dem Menü Daten über Position und Aktivierungsstatus des Eingabegeräts übermittelt werden. Aus diesen Daten entscheidet das Menü anschließend, ob ein Eintrag aktiviert oder eine Geste gezeichnet wurde.

3.3 Zusammenfassung

Das implementierte Marking-Menü Framework setzt soweit alle in Kapitel 3.1.: *Grundlagen* aufgestellten Anforderungen um und entspricht der Definition Kurtenbachs mit einigen Erweiterungen. Es bietet eine gute Grundlage für Umsetzungen nachfolgender Arbeiten. Die Frage der Umsetzung von User Interface Widgets zur numerischen Werteeingabe kann im Rahmen dieses Frameworks beantwortet werden.

4 Prototyp Implementierung

Aus den verwandten Arbeiten sollen zunächst Erkenntnisse und Anforderungen für die Entwicklung der Schieberegler gezogen und zu einem Konzept, bestehend aus Anforderungsanalyse und Papierprototypentwicklung, gebündelt werden. Ergebnisse einer Pilotstudie sollen erste Hinweise auf die Performance und Bedienbarkeit der erstellten Konzepte geben. Die anschließende Umsetzung integriert den aus der Pilotstudie als am vielversprechendsten hervorgegangenen Prototypen in das Marking-Menü Framework und visualisiert die Interaktion mit diesem.

4.1 Anforderungsanalyse

Damit eine klare Entwicklungsgrundlage geschaffen werden kann, soll vor der Entwicklung und Umsetzung der Prototypen eine Anforderungsanalyse durchgeführt werden. Ziel dieser Analyse ist die Schaffung einer Anforderungsübersicht mit Wichtung der Anforderungen von „obligatorisch“ bis „fakultativ“.

Wie in Kapitel 2.5.: *Schieberegler* beschrieben besteht das Element des Schiebereglers mindestens aus den Elementen Wertebereich und Indikator.

Eine **Funktionalität zur schrittweisen Veränderung** des Schiebereglerwerts wird durch Galitz [11] und Vora [35] beschrieben.

Aus der Arbeit von Gebhardt u. a. [13, S. 651] geht hervor, dass der aktuelle Wert, sowie das **Minimum und Maximum** des Wertebereichs des Schiebereglers **visualisiert** werden sollte.

> [...] current value of the slider as well as minimum and maximum values should be drawn [...]
>
> — Gebhardt u. a. [13, S. 651]

Die **Integration** eines Schiebereglers **in** das **Menü** soll nicht aus einer reinen Kombination der Elemente bestehen, sondern soll die Vorteile des Pie-Menüs mit dem Element des Schiebereglers vereinen. Dies erlaubt dem Anwender effizient mit dem Schieberegler zu interagieren.

Ein **Indikator** soll die aktuelle Position auf dem Wertebereich entlang der Achse anzeigen. Der Indikator kennzeichnet den aktuell ausgewählten Wert des Schiebereglers.

Die direkte **Manipulation des Indikators** soll möglich sein. Eine Bewegung des Indikators verändert zeitgleich den Wert des Schiebereglers.

Der aktuell eingestellte Wert des Schiebereglers soll klar erkennbar sein. Eine **Visualisierung des Werts** soll in den Schieberegler integriert werden.

Der **Wert** des Schiebereglers soll, ohne zusätzliche Aktionen ausführen zu müssen, **aus** dem **Menü** heraus **ablesbar** sein.

Die Änderungsrate des Schiebereglers soll durch eine mathematische Funktion beschrieben werden können. Neben einer linearen Änderungsrate soll zusätzlich die **Nichtlinearität** unterstützt werden. Die Unterstützung verschiedener Änderungsraten ermöglicht ein erweitertes Einsatzgebiet.

Eine **Skala** soll den Werteverlauf über den Wertebereich hinweg visualisieren. Die Skala visualisiert die Änderungsrate des Schiebereglers über die Länge des Wertebereichs und visualisiert die Extremwerte.

Der gesamte Wertebereich soll **Markierungen in** bestimmten **Intervallen** enthalten. Intervalle vermitteln die ungefähre Präzision des Schiebereglers.

Der Schieberegler soll **nach** einer **Interaktion geöffnet** bleiben. Dies erlaubt die Durchführung mehrerer aufeinanderfolgender Aktionen.

Das **Schließen** des Schiebereglers soll nicht automatisch, sondern **durch** eine **dedizierte Aktion** erfolgen. Ein automatisches Schließen des Schiebereglers nach der Durchführung einer Aktion vermindert die Performance und ist nicht wünschenswert.

Der Schieberegler soll **performant** sein, große Wertebereiche sollen schnell ausgewählt und erkundet werden können.

Das Design des Schiebereglers soll **übersichtlich** gestaltet werden. Eine visuelle Überladung ist zu vermeiden.

4.1.1 Klassifizierung und Zusammenfassung

Die Liste der aufgestellten Anforderungen lässt erkennen, dass manche zusammengefasst werden können, und andere aufeinander aufbauen. Unterteilt werden können die Anforderungen in die Kategorien funktional und nichtfunktional.

Funktionale Anforderungen entsprechen dem Funktionsumfang des Schiebereglers, die nichtfunktionalen Anforderungen beschreiben die Qualität des Schiebereglers.

Die Anforderung den Schieberegler nach einer durchgeführten Interaktion nicht automatisch zu schließen kann aus der Anforderung des Schließens durch eine dedizierte Aktion abgeleitet und unter dieser zusammengefasst werden.

Eine Wertigkeitsanzeige im Menü setzt die Umsetzung der Anforderung der Anzeige des aktuellen Werts voraus. Ohne eine Anzeige der Wertigkeit des Schiebereglers ist das Ablesen des Werts aus dem Menü nicht durchführbar. Beide Anforderungen können zu einer zusammengefasst werden.

Die direkte Manipulation des Indikators setzt voraus, dass ein Indikator in dem Schieberegler vorhanden ist.

Funktional	Nichtfunktional
Indikatoranzeige	Hohe Performance
Indikatormanipulation	Gute Übersichtlichkeit
Integration in Menü	
Nichtlinearität	
Schließen durch dedizierte Aktion	
Schrittweise Wertänderung	
Skala	
Anzeige des Schiebereglerwerts	

Tabelle 2: Zusammenfassung der Schieberegleranforderungen ohne Wichtung.

Die Anforderungen der Intervallmarkierungen und Anzeige des Minimal- und Maximalwerts sind in der Anforderung Skala implizit enthalten. Eine Skala enthält eine Visualisierung des Wertebereichs in definierten Abständen, dies entspricht den Intervallmarkierungen. Die Extremwerte des Schiebereglers entsprechen dem Start und Ende der Skala.

Die Anforderung „Integration in Menü“ ist wie die Nichtlinearität als eigenständig zu werten. Diese Anforderungen bauen nicht auf vorausgehenden Anforderungen auf oder können unter anderen zusammengefasst werden. Gleiches gilt für die nichtfunktionalen Anforderungen „Performance“ und „Übersichtlichkeit“.

Die kategorisierten und zusammengefassten Anforderungen sind in Tabelle 2 dargestellt.

4.1.2 Gewichtung der aufgestellten Anforderungen

Die zusammengefassten Anforderungen aus Kapitel 4.1.1.: *Klassifizierung und Zusammenfassung* sollen der Wichtigkeit nach aufgestellt und bewertet werden.

Integration in Menü Die Integration der Funktionalitäten des Pie-Menüs in den Schieberegler ist obligatorisch. Pie-Menüs bieten distinktive Vorteile in der Benutzung. Eine Übertragung dieser auf den Schieberegler erlaubt eine sinnvolle Integration des Widgets in die Menüstruktur.

Die reine Kombination eines Schiebereglers mit einem Marking-Menü wurde bereits durch McGuffin, Burtnyk und Kurtenbach [27] beschrieben.

Wertigkeitsanzeige Eine Umsetzung von Schiebereglern in Pie-Menüs durch Gebhardt u. a. [13, S. 651] zeigte, dass Probanden eine Visualisierung des aktuell geänderten Werts des Schiebereglers benötigen.

Die Anzeige des Schiebereglerwerts bietet zum einen den Vorteil klar zu kommunizieren, ob der gewünschte Wert erreicht wurde und zum anderen den Wert des Schiebereglers aus dem Menü heraus ablesen zu können, ohne das Widget aktivieren zu müssen.

Schließen durch dedizierte Aktion Die obligatorische Anforderung des Schließens durch eine dedizierte Aktion ermöglicht die Durchführung mehrerer aufeinanderfolgender Interaktionen in dem Schieberegler ohne, dass nach jeder Interaktion das Widget erneut aktiviert werden muss.

Indikatoranzeige Der Indikator des Schiebereglers vermittelt die ungefähre Position auf dem Verlauf des Wertebereichs. Durch diesen kann intuitiv und schnell der Bereich erkannt werden, in dem der Wert des Schiebereglers liegt. Für die Funktionalität eines Schiebereglers ist die Anzeige eines Indikators jedoch nicht erforderlich. Der in Abbildung 7 dargestellte Schieberegler visualisiert die Wertigkeit ohne dedizierten Indikator. Eine Indikation des Werts auf dem Wertebereich ist somit als wünschenswert, aber nicht obligatorisch zu bewerten.

Skala Probanden der Studie von Gebhardt u. a. [13, S. 651] beschrieben, dass die Visualisierung des Wertebereichs des Schiebereglers unterstützend in der Bedienung wirken würde. Das Element der Skala verbindet die Anforderungen der Intervallmarkierung und Visualisierung der Extremwerte.

Die Umsetzung der Skala ist in der Funktionalität des Schiebereglers nicht obligatorisch, wirkt aber unterstützend während der Anwendung und ist somit wünschenswert in der Umsetzung.

Übersichtlichkeit Der Schieberegler sollte übersichtlich und intuitiv gestaltet sein. Eine unübersichtliche Gestaltung kann zu einer verminderten Performance und Unverständlichkeit des Schiebereglers führen.

In der erstmaligen Benutzung des Schiebereglers sollte die Funktionalität intuitiv klar sein und keine Erklärungen benötigen. Während der Entwicklung sollte die Übersichtlichkeit beachtet werden.

Performance Große Wertebereiche sollten schnell und performant in dem Schieberegler erkundet werden können. Gegensätzlich müssen zusätzlich kleine Wertebereiche gut durch den Schieberegler auswählbar gemacht werden. Die annehmbaren Wertebereiche und Schrittgrößen sind stark abhängig von der Art des Schiebereglers, die Anforderung der Performance ist wünschenswert.

Schrittweise Wertänderung Wie durch Harley [15] beschrieben, ist das Element des Schiebereglers besonders in Anwendungsfällen geeignet, in denen ein ungefährer, statt eines exakten Werts, benötigt wird.

Die Funktion der schrittweisen Wertänderung wird bei Schiebereglern mit geringem Schrittabstand benötigt oder in Anwendungsfällen bei denen ein exakter Wert ausgewählt werden muss. In der Umsetzung ist die Anforderung der schrittweisen Wertänderung als fakultativ zu werten.

Obligatorisch	Integration in Menü
	Wertigkeitsanzeige
	Schließen durch dedizierte Aktion
Wünschenswert	Indikatoranzeige
	Skala
	Übersichtlichkeit
	Performance
Fakultativ	Schrittweise Wertänderung
	Indikatormanipulation
	Nichtlinearität

Tabelle 3: Gewichtung der Schieberegleranforderungen von „obligatorisch“ bis „fakultativ“

Indikatormanipulation Durch die Manipulation des Indikators kann performant ein ungefährer Wert auf dem Wertebereich des Schiebereglers ausgewählt werden. Der Wert eines Schiebereglers muss jedoch nicht exklusiv über die Manipulation des Indikators geschehen. Denkbar ist beispielsweise die Bewegung der gesamten Skala statt des Indikators. Die Anforderung der Indikatormanipulation kann als fakultativ bewertet werden.

Nichtlinearität Der Großteil der verwendeten Schieberegler besitzt eine lineare Änderungsrate. Die Beschreibung der Änderungsrate über eine mathematische Funktion erweitert das Einsatzgebiet des Schiebereglers. Aufgrund des geringen Einsatzes nichtlinearer Schieberegler ist diese Anforderung als fakultativ zu bewerten.

4.1.3 Fazit

Eine zusammengefasste Aufteilung und Gewichtung der Anforderungen von „obligatorisch“ bis „fakultativ“ ist in Tabelle 3 gegeben.

Obligatorische Anforderungen sind solche, die durch jeden Schieberegler umgesetzt werden müssen. Diese Anforderungen stellen den grundlegenden Funktionsumfang eines Schiebereglers dar. Die Integration des Schiebereglers in die Bedienvorteile des Pie-Menüs wurde als am wichtigsten gewertet. Aus der Bedienung eines Pie-Menüs geht die Anforderung des dedizierten Schließens hervor. Diese ist ebenso wichtig wie die Anzeige des Schiebereglerwerts.

Anforderungen der Kategorie Wünschenswert sind solche Anforderungen, die unterstützende Funktionen beinhalten, für den grundlegenden Funktionsumfang aber nicht zwingend umgesetzt werden müssen.

Die letzte Kategorie der fakultativen Anforderungen enthält Funktionalitäten, die ihre Anwendung in bestimmten Schiebereglern finden oder eine Hilfestellung während der Bedienung darstellen. Sie erweitern den Funktionsumfang eines Schiebereglers, werden jedoch nicht zwingend benötigt.

4.2 Prototypen

Ausgehend von den aufgestellten Anforderungen erfolgte die Entwicklung verschiedener Prototypen. Diese werden nachfolgend beschrieben und in Hinblick auf die erstellten Anforderungen analysiert.

4.2.1 Circleslider

Die Idee des Circleslider ist, die Wertigkeit des Schiebereglers über einen veränderbaren Kreisradius abzubilden. Durch Halten und Ziehen des Kreises wird die Wertigkeit des Schiebereglers eingestellt, eine Bewegung vom Kreiszentrum weg erhöht den Wert, eine Bewegung in entgegengesetzte Richtung verringert diesen. Die Veränderung des Indikationskreises ist nicht auf eine bestimmte Richtung beschränkt, sondern kann in jede Richtung erfolgen.

Die Veränderung des Indikators aktualisiert zeitgleich den angezeigten Wert im Menüelement. Die Aktivierung des Eingabegeräts im Anzeigebereich des Schiebereglers verändert die Größe des Kreises bis zu der Position des Eingabegeräts.

Die Beendigung der Interaktion erfolgt durch Aktivierung des Eingabegeräts im Zentrum des Menüelements.

Der Prototyp des Circleslider ist in Abbildung 12 dargestellt. Die Einbindung des Schiebereglers in das Marking-Menü erfolgt über die Anzeige als normales Menüelement. Die Navigation in den Schieberegler behält die Referenz zum Elternelement, in dem dieses weiter halbtransparent angezeigt wird.

Der aktuell eingestellte Wert des Schiebereglers wird mittig im Menüelement dargestellt und ist zusätzlich statt eines Icons vgl. Abbildung 10 (b) aus dem Menü heraus ablesbar. Nach einer durchgeführten Änderung des Werts wird der Schieberegler nicht automatisch geschlossen, eine Navigation zurück in das Elternelement erfolgt durch das Klicken auf das Menüelement des Schiebereglers. Eine zusätzliche Rücknavigation kann durch den in Kapitel 3.2.4.: *Interaktion* beschriebenen Rücknavigationssektor erfolgen.

Der Circleslider erfüllt alle beschriebenen obligatorischen Anforderungen grundlegend. Nachteilig ist die Benutzung des Schiebereglers jedoch während der Anwendung des Marking-Modus. Werte können nicht fließend durch eine Geste ausgewählt werden, sondern müssen durch dedizierte Interaktionen eingestellt werden. Die Integration des Schiebereglers während der Klicknavigation gelingt aufgrund der ohnehin ausgeführten dedizierten Eingaben somit besser.

Die vier als wünschenswert klassifizierten Anforderungen aus Tabelle 3 werden durch den Circleslider in Gänze abgedeckt. Die Anforderung der Indikatoranzeige sowie der fakultativen Indikatormanipulation werden durch den in Abbildung 12 (d) gezeigten Indikator erfüllt.

Der Wertebereich des Schiebereglers wird in bestimmten Abständen durch halbtransparente Kreise dargestellt, dies erfüllt die Anforderung der Skala und der Performance. Die Veränderung des Werts durch Klick auf der Skala ermöglicht eine performante Bedienung.

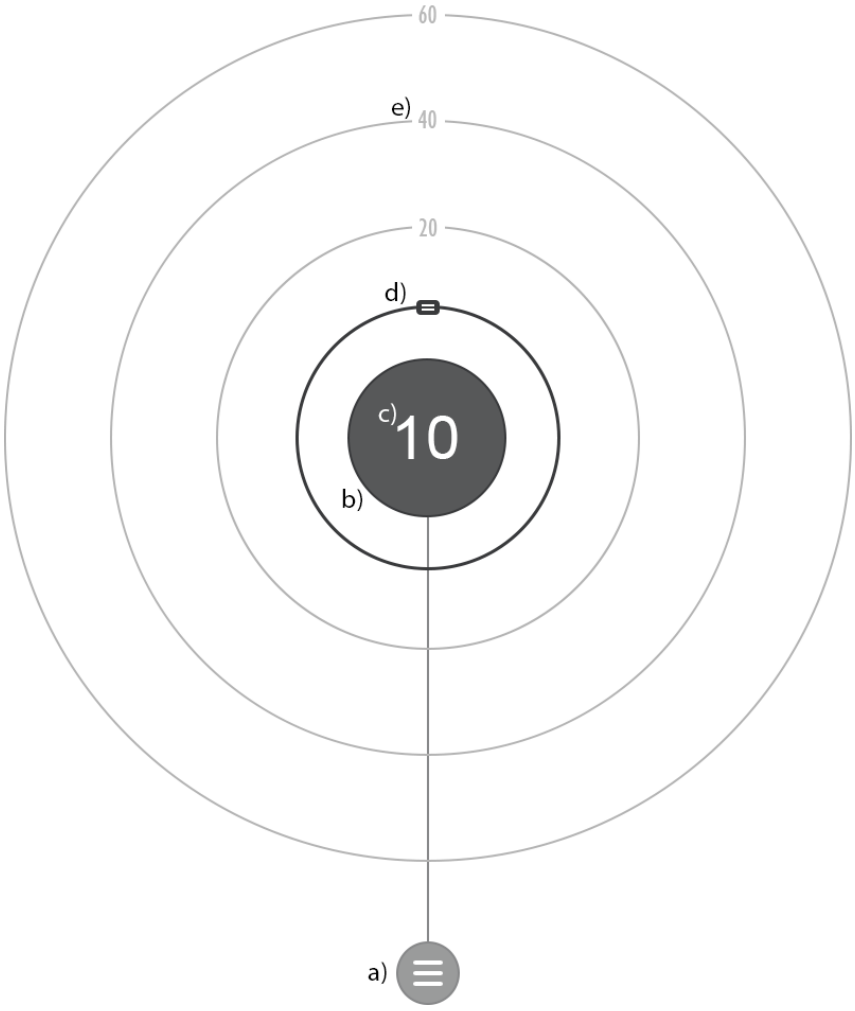


Abbildung 12: Prototyp des Circleslider. Die Abbildung zeigt den Schieberegler nach der Öffnung aus dem Hauptmenü. Die Referenz des Elternelements ist unter (a) abgebildet. Das Menüelement des Schiebereglers ist in (b) zu erkennen. In diesem wird die aktuelle Wertigkeit angezeigt (c). Der bewegliche Indikator ist in (d) angezeigt. Die Skala des Schiebereglers (e) ist durch halbtransparente Kreise abgebildet. Quelle: *Eigene Darstellung.*

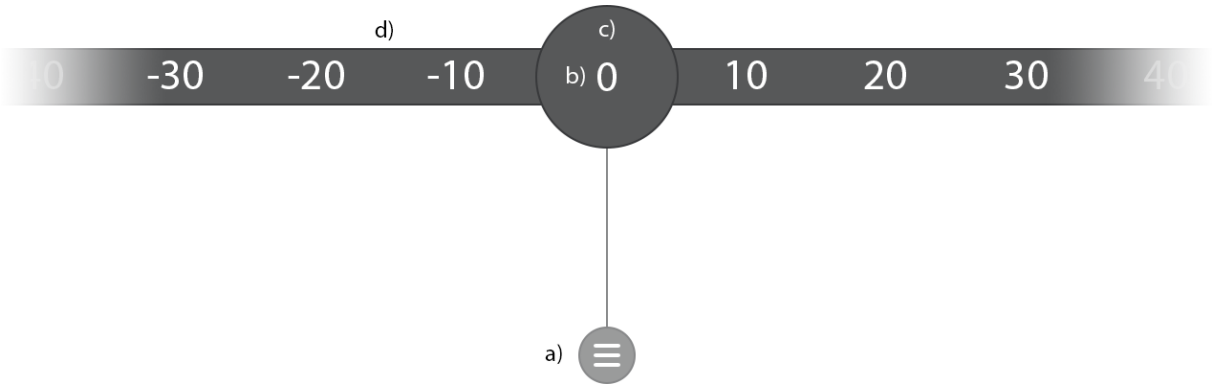


Abbildung 13: Prototyp des Ribbonslider. Das Elternelement (a) des Schiebereglers ist halbtransparent dargestellt. Der aktuelle Wert (b) wird im Menüelement (c) des Schiebereglers angezeigt, dieser wird zeitgleich mit der Bewegung des Zahlenbands aktualisiert. Die annehmbaren Werte werden über das Zahlenband (d) auswählbar gemacht. Quelle: *Eigene Darstellung.*

Eine schrittweise Wertänderung ist durch die gegebenen Elemente des Schiebereglers nicht umsetzbar. Die Anforderung der Nichtlinearität fand während der Entwicklung des Prototyps keine Beachtung, ist aber trivial in den Schieberegler integrierbar.

Der Schieberegler ist nicht für den Einsatz in platzbegrenzten Anwendungen geeignet, große Wertebereiche benötigen viel räumlichen Platz. Dem Problem kann durch eine Verringerung der Abstände der Skala entgegengewirkt werden. Dies verringert im Gegenzug die Präzision des Schiebereglers.

Die eindimensionale Interaktionsebene des Schiebereglers ist nicht vorteilhafter durch andere Eingabegeräte zu bedienen. Im Bereich der Touchinteraktion ist eine Abbildung der Quetsch- und Spreizgesten auf die Veränderung der Größe des Indikatorkreises denkbar.

4.2.2 Ribbonslider

Der Ribbonslider visualisiert den Wertebereich eines Schiebereglers über ein Zahlenband. Das Band ist frei beweglich, der aktuelle Wert ist mittig im Menüelement des Schiebereglers angezeigt. Durch Ziehen und Halten des Eingabegeräts wird das Band bewegt, bis der gewünschte Wert erreicht ist.

Die Beendigung der Interaktion erfolgt neben der Aktivierung des Eingabegeräts im Zentrum des Menüelements zusätzlich über das durch das Menü hinzugefügte Rücknavigationselement.

Abbildung 13 zeigt den Prototypen des Ribbonslider.

Gleich dem Circleslider setzt der Ribbonslider alle obligatorischen Anforderungen um. Die Kategorie der wünschenswerten Anforderungen wird bis auf die Performance in Gänze abgedeckt.

Über die Auswahl eines Werts auf dem Zahlenband kann performant ein Wert ausgewählt werden, die Maximal- und Minimalwerte des Schiebereglers können im Normalfall nicht sofort ausgewählt werden. Dies erzeugt Probleme in Anwendungsfällen großer Zahlenbereiche. Zwar kann die Skala so konfiguriert werden, dass der gesamte Zahlenbereich sichtbar ist und ausgewählt werden kann, dies verringert im Gegenzug aber die Präzision des Schiebereglers.

Die Indikatoranzeige ist über das Menüelement des Schiebereglers abgedeckt, das Element entspricht dem Indikator des Schiebereglers. Das bewegliche Zahlenband setzt die Anforderung der Skala um.

Aus den drei fakultativen Anforderungen setzt der Prototyp die Anforderung der Nichtlinearität um. Werte des Schiebereglers können durch verschiedene mathematische Funktionen anhand des Zahlenbands visualisiert werden. Das Zahlenband ist nicht platzlimitiert und kann jegliche Zahlenbereiche annehmen. Die Anforderung der schrittweisen Wertänderung fand keine Umsetzung in dem Prototypen. Das Menüelement des Schiebereglers verweilt über die Bediendauer statisch an der Stelle der Öffnung, es dient als Indikator und kann nicht bewegt werden.

Vorteil des Schiebereglers ist die Anwendung in großen Zahlenbereichen mit entsprechend breit konfigurierter Skala. Das Zahlenband erlaubt beidseitige „unendliche“ Bewegungen, der Zahlenbereich ist nicht limitiert. Während der Interaktion mit dem Schieberegler könnte die Bewegungsgeschwindigkeit des Eingabegeräts betrachtet werden. Eine schnelle Bewegung kann zu großen Wertänderungen führen, eine langsame Bewegung zu sehr kleinen.

Toucheingaben erweitern die Bedienungsmöglichkeiten des Schiebereglers durch Gesten. Verschiedene Eingaben können auf unterschiedliche Funktionen gelegt werden. Möglich wäre beispielsweise die Veränderung der Schrittgröße durch eine Spreiz- bzw. Quetschgeste oder der Sprung zu den Extremwerten über eine Schnipsgeste. Die Bewegung des Zahlenbandes erfolgt über die direkte Manipulation auf dem Bildschirm.

Der Einsatz eines Trackballs könnte in der Benutzung performanter sein als die Interaktion mit der Computermaus. Die Verbindung der Bewegung des Trackballs mit der Verschiebung des Zahlenbands bietet den Vorteil nicht räumlich begrenzt zu sein. Wo die Computermaus physikalisch in der Größe der Bewegung limitiert ist, kann der Trackball stationär ins „unendliche“ bewegt werden.

4.2.3 Crankslider

Bei dem Prototypen des Crankslider werden Werte über Kurbelbewegungen eingestellt. Rechtsdrehende Bewegungen erhöhen den Wert des Schiebereglers, linksdrehende Bewegungen verringern die Wertigkeit. Ein Indikator in Kugelform zeigt die relative Position auf dem Wertebereich an.

Die Bewegung ist nicht auf eine Umdrehung begrenzt, sondern kann „unendlich“ lang durchgeführt werden. Die Einstellung des Schiebereglers determiniert, ob der Wert nach einer bestimmten Anzahl Umdrehungen stoppt, der Wertebereich also begrenzt ist.

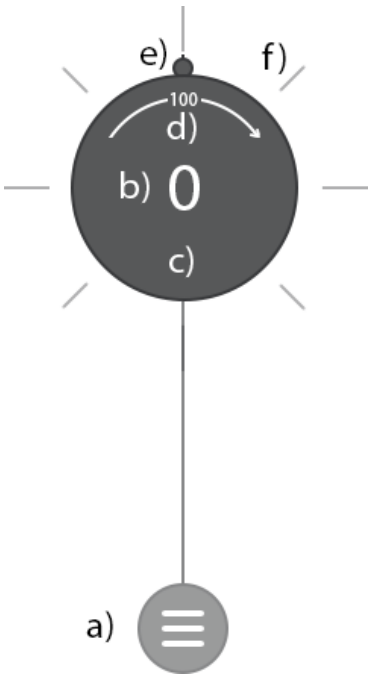


Abbildung 14: Prototyp des Crankslider. Eine Referenz zu dem Elternelement (a) ist durch die Anzeige über eine Halbtransparenz dargestellt. Der Wert des Schiebereglers (b) wird in dem Menüelement (c) mittig angezeigt und wird zeitgleich mit der Veränderung des Werts aktualisiert. Die Größe der Wertänderung einer Umdrehung wird in (d) durch einen rechtsdrehenden Pfeil visualisiert. Eine Umdrehung des Prototypen entspricht einer Wertänderung von 100. Der Indikator des Schiebereglers ist in (e) gekennzeichnet. Eine wertfreie Skala (f) unterteilt den Wertebereich in feste Schritte. Quelle: *Eigene Darstellung.*

Der in Abbildung 14 dargestellte Prototyp des Crankslider erfüllt alle als obligatorisch gewerteten Anforderungen. Die Wertigkeit des Schiebereglers ist zu jeder Zeit aus dem Marking-Menü heraus erkennbar. Die Integration in das Marking-Menü erlaubt es nach einer Navigation in den Schieberegler, den Wert ohne Zeitverzug zu verändern. Eine Wertänderung kann innerhalb einer Geste erfolgen.

Über den angezeigten Indikator kann der Wert des Schiebereglers verändert werden. Dies deckt die Anforderungen Indikatoranzeige und Indikatormanipulation teilweise ab, da der Indikator nicht den aktuell eingestellten Wert, sondern nur eine Position auf dem Wertebereich des Menüelements anzeigt. Die Aktivierung des Eingabegeräts in einem Bereich der Skala platziert den Indikator auf die entsprechende Position.

Die Performance des Schiebereglers leidet im Bereich großer Zahlen. Die Auswahl der Maximal- und Minimalwerte kann nicht direkt geschehen, die Werte müssen durch die entsprechende Anzahl Rotationen ausgewählt werden.

Eine Schrittweise Wertänderung ist in dem Schieberegler nicht trivial umzusetzen, die gegebenen Elemente sind für diese Anforderung nicht erweiterbar.

Nichtlineare Werte können über den Schieberegler abgedeckt werden, die Art der Wertänderung kann in dem Menüelement angezeigt werden vgl. Abbildung 14 (d).

Toucheingaben führen in der Bedienung des Schiebereglers zu keinen Performancevorteilen gegenüber der Maus. Beide Eingabemethoden müssen eine Kreisbewegung für die Veränderung des Werts durchführen.

Der Einsatz eines Gamepads könnte in der Bedienung des Crankslider Performancevorteile bieten. Kreisbewegungen können über die an den Controllern vorhandenen Steuerknüppel schnell ausgeführt werden. Gleiches gilt für den Einsatz einhändig gehaltener VR-Controller, die eine berührungsempfindliche Oberfläche besitzen. Eine Kreisbewegung kann auf dieser Oberfläche performant mit dem Daumen ausgeführt werden.

4.2.4 Morphslider

Werte des Morphslider werden über die veränderbare Form des Schiebereglers angepasst. Der Schieberegler besteht aus einer dreiseitig veränderbaren Kugelform. Zwei Seiten dienen der schrittweisen Änderung des Schiebereglerwerts, die dritte Seite verändert durch Halten und Ziehen die Wertigkeit analog der Position des Indikators.

Abbildung 15 stellt den Prototypen grafisch dar.

Analog der drei beschriebenen Schieberegler deckt der Morphslider alle obligatorischen Anforderungen ab. Dedizierte Aktionen, Nutzung des Rücknavigationssektors oder Aktivierung des Eingabegeräts in dem Menüelement, beenden die Interaktion mit dem Schieberegler. Die Wertigkeit wird in dem aktiven Modus mittig im Menüelement visualisiert, im deaktivierten Modus zeigt der Schieberegler den Wert statt eines Menüicons an.

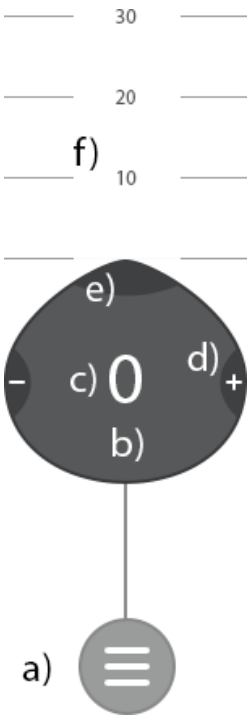


Abbildung 15: Prototyp des Morphslider. Die Referenz zum Elternelement wird in (a) dargestellt, das Menüelement des Schiebereglers unter (b). Der Menüeintrag enthält vier Elemente zu der Bedienung des Schiebereglers. Die Wertigkeit des Schiebereglers (c) wird mittig als Zahl in dem Element angezeigt. Farblich hervorgehobene Stellen markieren Griffpunkte, diese werden durch Halten und Ziehen bedient. Die Funktionalität zur schrittweisen Änderung des Werts erfolgt über die seitlichen Griffpunkte (d). Der Indikator des Schiebereglers (e) ist frei beweglich und kann auf eine gewünschte Werteposition der Skala (d) bewegt werden. Die Wertigkeit des Schiebereglers legt die Streckung der Kugelhälfte des Indikators fest. Quelle: *Eigene Darstellung.*

Der Schieberegler wird als normales Menüelement in das Marking-Menü integriert, die Richtung des Indikators entspricht dem zugewiesenen Winkel des Schiebereglers.

Abbildung 15 zeigt den Morphslider mit einem eingestellten Winkel von 0°, eine Einstellung von 180° würde einer Spiegelung an der x-Achse entsprechen.

Der wünschenswerte Indikator wird durch eine spitz zulaufende Seite dargestellt, diese kann direkt manipuliert und durch das Eingabegerät bewegt werden. Die Position des Indikators entspricht der Wertigkeit des Schiebereglers auf der angezeigten Skala. Wenige Elemente verschaffen dem Schieberegler eine Übersichtlichkeit, die Maximal- und Minimalwerte können performant ausgewählt werden.

Die fließende Bewegung und Veränderung des Indikators während des Marking-Modus ist nicht trivial umzusetzen, die Veränderung des Indikators kann nur sinnvoll über die Klicknavigation geschehen. Die Anforderung der vollständigen Integration in das Marking-Menü ist nicht in Gänze abgedeckt.

Zwei dedizierte Seiten des Morphslider dienen der schrittweisen Änderung des Schiebereglerwerts. Durch Klicken oder Ziehen und Halten kann der Wert des Schiebereglers angepasst werden. Die Länge der Verformung einer Seite entspricht der Geschwindigkeit der schrittweisen Wertänderung. Während des Marking-Modus kann die Verformung entsprechend der Richtung der ausgeführten Geste erfolgen.

Nichtlineare Werte waren nicht Teil der Betrachtung während der Entwicklung des Prototypen. Die Nichtlinearität kann jedoch trivial über die Anzeige einer nichtlinearen Skala umgesetzt werden.

Die Präzision des Schiebereglers nimmt mit zunehmender Größe des Zahlenbereichs ab. Große Zahlenbereiche benötigen für präzise Eingaben mehr Anzeigefläche.

Eingabegeräte mit frei belegbaren Tasten könnten während der Anwendung der schrittweisen Wertänderung performanter gegenüber Maus oder Toucheingaben sein. Die Änderung des Werts über dedizierte Tasten erlaubt den Wert über die Verformung des Schiebereglers einzustellen und anschließend eine präzise Auswahl zu treffen, ohne das Eingabegerät bewegen zu müssen.

4.3 Ergebnisse der Papierprototypen

Die entwickelten Prototypen wurden in einer Pilotstudie mit acht Personen untersucht und in den Kategorien Verständlichkeit, Spaß und Performance bewertet.

4.3.1 Pilotstudie

In einer durchgeführten Pilotstudie wurden acht Testpersonen, im Alter zwischen 18 und 40 Jahren, die in Anhang A dargestellten Prototypen in zufälliger Reihenfolge vorgelegt. Die Liste der fünf Prototypen umfasst neben der vier Schieberegler aus Kapitel 4.2.: *Prototypen* zusätzlich den in Kapitel 2.5.3 beschriebenen FaST Slider.

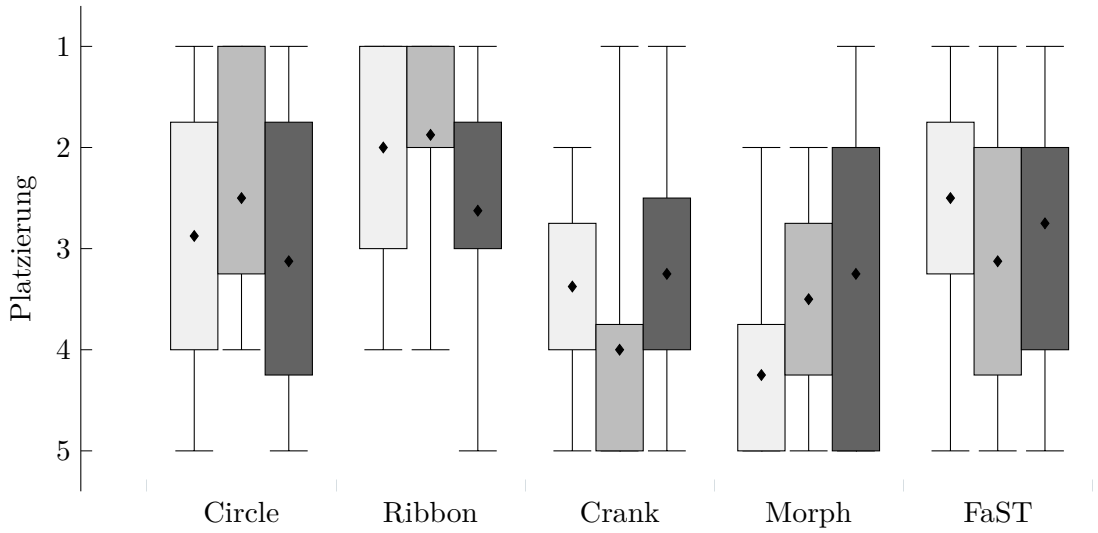


Abbildung 16: Auswertung der Pilotstudie. Platzierung der Prototypen anhand der drei Kategorien *Verständlichkeit* □, *Spaß* ▫ und *Performance* ■. Die Antennen zeigen die kleinsten und größten Datenwerte an. Der Mittelwert ist durch einen Diamanten dargestellt. Das 25% und 75% Quartil entspricht den angezeigten Boxen.

Vor dem Beginn eines Testgangs erhielt jeder Proband eine Einführung in das Thema der Arbeit und in die Art der Bedienung eines Pie-Menüs. Ein Zufallszahlengenerator bestimmte vor der Durchführung jedes Tests die Reihenfolge der vorgelegten Schieberegler. Die Bildreihenfolge einer Schiebereglergruppe war fest definiert. Alle getesteten Gruppen sind in Anhang A angezeigt.

Die Aufgabenstellung jeder Testperson war, einen bestimmten Wert in jedem vorgelegtem Schieberegler einzustellen und die durchgeführte Interaktion detailliert zu beschreiben. Beschrieben werden sollte der Start der Interaktion mit Position und Status des Eingabegeräts, die durchgeführte Bewegung zu der Änderung des Schiebereglerwerts sowie die Erwartung der Reaktion des Schiebereglers auf die Änderung des Werts.

Nach der Beendigung der gestellten Aufgaben wurde den Probanden der Endzustand des Menüs vorgelegt.

Kommentare der Testpersonen wurden während der Durchführung notiert und dienen als Grundlage für folgende Schiebereglerrevisionen.

Jeder Teilnehmer gab vor Ende des Tests jedem Prototypen eine Platzierung zwischen 1 und 5 in den Kategorien Verständlichkeit, Spaß und Performance. Die Bilder der Prototypen nach initialer Aktivierung wurden den Probanden in zufälliger Reihenfolge vorgelegt. Die Kategorien wurden der Reihenfolge nach beschrieben und die Bewertungen abgegeben. Platz 1 entspricht dem besten, Platz 5 dem schlechtesten, jeder Prototyp kann in einer Kategorie nur eine Platzierung erhalten. Es ist somit nicht möglich, dass ein Schieberegler dieselbe Platzierung mit einem anderen Schieberegler in einer Kategorie teilt.

4.3.2 Ergebnis

Die Daten der durchgeführten Pilotstudie sind in Abbildung 16 als Kastengrafik visualisiert. Jeder Prototyp bildet eine Gruppe aus den drei Kategorien Verständlichkeit, Spaß und Performance.

Ersichtlich ist, dass der Morphslider mit einem Mittelwert von 4.3 in der Kategorie Verständlichkeit auf dem letzten Platz liegt. Dies deckt sich mit den Kommentaren der Probanden während der Tests, in sieben von acht Durchgängen wurde eine Erklärung über die Bedienung des Morphslider benötigt.

Am schlechtesten bewertet ist der Crankslider mit einem Mittelwert von 4 in der Kategorie Spaß. Bemerkungen der Probanden gingen auf die problematische Handhabung des Schiebereglers bei der Eingabe großer Zahlenwerte ein.

Der Circleslider und der FaST Slider liegen in der Bewertung annähernd gleichauf. Der Circleslider wurde durch die Probanden als spaßiger in der Bedienung bewertet als der FaST Slider, Mittelwert 2.5 gegenüber 3.1. Die Performance des Circleslider ist mit 3.1 schlechter bewertet als die des FaST Slider (2.8).

Aus den Daten kann der Ribbonslider als spaßigster Schieberegler gewertet werden. Der Großteil aller Bewertungen liegt im Bereich 1 und 2. Die Interaktion mit dem Schieberegler war für jeden Probanden ohne Erklärung ersichtlich. In der Kategorie Performance liegt der Mittelwert des Ribbonslider mit 2.5 gleichauf mit dem FaST Slider.

Insgesamt schneiden die Versionen Morph- und Crankslider am schlechtesten in den Bewertungen Verständlichkeit und Spaß unter den fünf getesteten Schiebereglern ab. Sie scheiden somit für eine weitere Betrachtung aus. Aufgrund der positiveren Bewertung soll der Ribbonslider zunächst unter Betrachtung der abgegebenen Kommentare neu analysiert werden.

Das Ergebnis sollen anschließend für eine neue Revision des Schiebereglers genutzt und umgesetzt werden. Die Schieberegler Circleslider und FaST Slider sollen in einer zukünftigen Arbeit umgesetzt und mit dem Ribbonslider verglichen werden.

4.4 Umsetzung

Der Prototyp des Ribbonslider soll anhand des Feedbacks der Pilotstudie überarbeitet werden. Kommentare der Probanden werden zu Beginn beschrieben und klassifiziert. Anschließend werden Änderungen an dem überarbeiteten Schieberegler dargestellt und mit den neu aufgestellten Anforderungen aus der Pilotstudie verglichen. Damit der Interaktionsumfang des Schiebereglers dargestellt werden kann, erfolgt eine abschließende Umsetzung in dem Paper.js Framework [22].

4.4.1 Nutzerfeedback

Nach Vorlage der zwei verschiedenen Visualisierungen des endenden Wertigkeitbereichs präferierten 75% der Testpersonen die Darstellung des Zahlenbandes ohne transparent auslaufende Enden vgl. A.2.: *Ribbonslider* Bild 2 – 3.

Hieraus kann die Anforderung abgeleitet werden, das Ende des Zahlenbereichs ohne Transparenz anzuzeigen.

Fünf von acht Personen versuchten während der Aufgabe den Schiebereglerwert von 0 auf 20 zu ändern, den Indikator zu bewegen und nicht wie vorgesehen das Zahlenband. Die Bewegung erfolgte von links nach rechts anstatt von rechts nach links.

Es ergeben sich zwei Anforderungen aus dieser Beobachtung. Der Indikator des Schiebereglers muss als nicht bewegliches Objekt dargestellt werden, zusätzlich muss das Zahlenband als beweglich visualisiert werden.

Während der Benutzung versuchten drei Probanden die Wertigkeit des Schiebereglers durch direkte Auswahl des Werts einzustellen. Aus dieser Interaktion kann die Anforderung abgeleitet werden, dass Zahlen des Schiebereglers auswählbar gemacht werden müssen. Dies deckt sich mit der fakultativen Anforderung aus Kapitel 3.: *Gewichtung der Schieberegleranforderungen* „Indikatormanipulation“. Statt der Bewegung des Indikators wird das Zahlenband auf die entsprechende Position bewegt.

Aus den acht Teilnehmern wünschten 25% während der Anwendung eine Funktionalität zur schrittweisen Änderung des Schiebereglerwerts. Diese Funktionalität fand in der ersten Revision des Prototypen keine Beachtung.

Tabelle 3 bewertet die Anforderung der schrittweisen Wertänderung als fakultativ, ungeachtet dessen soll diese Funktionalität in der neuen Schiebereglerrevision beachtet werden.

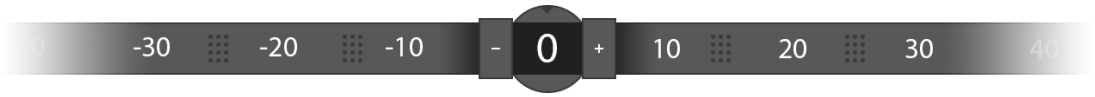
4.4.2 Überarbeitung des Schiebereglers

Die Anforderung die Enden der Zahlenbereiche ohne eine Transparenz anzuzeigen kann trivial umgesetzt werden. Die in A.2.: *Ribbonslider* dargestellten Papierprototypen zeigen die Implementierung.

Aus der in Kapitel 4.2.2.: *Ribbonslider* beschriebenen Idee des Ribbonslider geht hervor, dass Werte des Schiebereglers durch die Bewegung des Zahlenbandes geändert werden sollen. Tests der beschriebenen Pilotstudie zeigten konträr, dass 60% der Anwender eine Wertänderung über die Bewegung des Indikators durchführten. Dies ist eine Abweichung von dem erdachten Design und somit als problematisch zu bewerten. Aus einer durchgeführten Online-Umfrage [28] gingen verschiedene Lösungsansätze hervor.

Die erste Revision des Schiebereglers stellt den Indikator fälschlicherweise in den Mittelpunkt der Interaktion, die überproportionale Darstellung lässt das Zahlenband in den Hintergrund rücken. Der Eindruck über die Beweglichkeit des Indikators wird zusätzlich durch die Darstellung über dem Zahlenband verstärkt.

(a) Überarbeitet Version ohne schrittweise Wertänderung



(b) Überarbeitet Version mit Funktion zur schrittweisen Änderung des Werts

Abbildung 17: Ribbonslider Revision 2

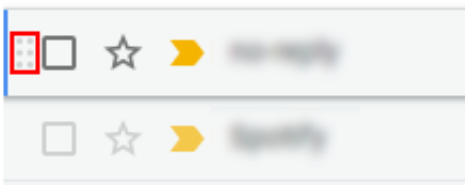
Das Zahlenband vermittelt nicht den Eindruck beweglich zu sein, eine Repräsentation der „Greifbarkeit“ wird nicht durch den entwickelten Prototypen umgesetzt.

Abbildung 17a und Abbildung 17b zeigen die überarbeitete Version des Schiebereglers. Damit die Interaktion mit dem Zahlenband in den Vordergrund rückt, erfolgte neben der Verkleinerung des Indikators zusätzlich eine Positionierung hinter dem Zahlenband. Dies stellt klar heraus, dass die Position des Indikators statisch ist und nicht bewegt werden kann.

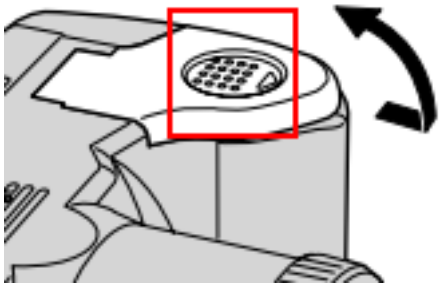
Dem Zahlenband wurden zusätzliche „grip points“ hinzugefügt. Diese Bereiche visualisieren die Greifbarkeit und Beweglichkeit des Zahlenbands. Beispiele des Einsatzes dieser Griffpunkte in anderen Anwendungen sind in Abbildung 18a und Abbildung 18b dargestellt.

Abbildung 17b zeigt eine alternative Darstellung des Ribbonslider mit Knöpfen zur schrittweisen Änderung des Schiebereglerwerts. Die Funktionalität kann nach Bedarf aktiviert und deaktiviert werden. Eine Aktivierung des Eingabegeräts auf einem der Knöpfe verändert die Wertigkeit des Schiebereglers schrittweise, das Zahlenband wird in die entsprechende Richtung verschoben.

Die aktuelle Wertigkeit des Schiebereglers verweilt mittig in dem Indikator, um eine Überlagerung des Schiebereglerwerts mit Zahlen des Zahlenbandes zu verhindern, verdeckt ein schwarztransparenter Verlauf das Zahlenband im Bereich des Indikators. Das Zahlenband wird im Bereich des Indikators verdeckt, der Indikator zeigt den veränderten Wert des Schiebereglers.



(a) Griffpunkte der Google Mail Weboberfläche (roter Kasten). Quelle: *Eigener Screenshot.*



(b) Griffpunkte einer Batterieabdeckung (roter Kasten). Quelle: Fuji Photo Film (Europe) G.m.b.H. [10, S. 13].

Abbildung 18: Umsetzung der Griffpunkte in verschiedenen Anwendungen.

4.4.3 Implementierung

Anhand der Überarbeitungen des Prototypen kann eine Umsetzung des Ribbonslider im Paper.js Framework erfolgen.

Eine Aufteilung des Schiebereglers fand in zwei Strukturen statt. Die grafische Struktur enthält alle zu zeichnenden Elemente des Schiebereglers. In der programmatischen Struktur sind die Logik und die Verarbeitung der Daten des Eingabegeräts enthalten.

Grafische Elemente Der Szenengraph des Ribbonslider enthält insgesamt vier Ebenen auf denen Elemente gezeichnet werden. Die oberste Ebene enthält den aktuell eingestellten Wert des Schiebereglers. Unter dieser Ebene liegt der schwarz-transparente Verlauf, der das Zahlenband verdeckt. Die vorletzte Ebene enthält das Element des Zahlenbands sowie die Griffpunkte und Wertschritte. Das kreisrunde Menüelement ist auf der untersten Ebene angezeigt und liegt visuell hinter dem Zahlenband.

Organisiert sind die Elemente in verschiedenen Gruppen. Die Gruppen-Klasse wird durch das Paper.js Framework bereitgestellt. Einer Gruppe kann ein bestimmter Blendmodus hinzugefügt werden, der entscheidet wie die Elemente untereinander vermischt werden. Diese Funktionalität wird genutzt, um die in Abbildung 17a dargestellten transparent auslaufenden Seiten umzusetzen.

Der gesamte Szenengraph des Schiebereglers ist in Abbildung 19 dargestellt. Das oberste Element entspricht der Klasse des Ribbonslider, diese bündelt alle enthaltenen grafischen Elemente in der „Geometry Group“. Diese Gruppe enthält an erster Stelle, bzw. unterster Ebene, das kreisförmig dargestellte Menüelement. Über diesem Element ist die „RibbonMask Group“ eingefügt, nachfolgend ist der schwarz-transparente Verlauf. An letzter Stelle wird der Wert des Schiebereglers als Textelement eingefügt.

Die transparent auslaufenden Seiten des Zahlenbandes werden durch den Einsatz zweier verschiedener Blendmodi umgesetzt. Die transparente Maskengruppe erhält den Blendmodus „source-over“. Dieser Modus zeichnet die Quelle (Geometriegruppe) über das Ziel (Maskengruppe), die Quelle ist überall dort sichtbar, wo das Ziel keine Pixel besitzt. Ohne Einsatz dieses Blendmodus wird das Elternmenü nicht angezeigt.

Innerhalb der Maskengruppe ist der transparent auslaufende Verlauf enthalten, sowie die „Ribbon Group“, die das Zahlenband und die Elemente des Bandes enthält. Die Zahlenbandgruppe besitzt den Blendmodus „source-in“, dieser Modus zeichnet die Quelle (Zahlenband) dort, wo der Hintergrund nichttransparente Pixel besitzt. Der transparent auslaufende Verlauf agiert somit als Maske für das gesamte Zahlenband.

Die Zahlenbandgruppe enthält drei grafische Elemente sowie $1 - n$ Griffpunktgruppen. An erster Stelle der Gruppe liegt das Element des Zahlenbandes, dies ist ein einfaches Rechteck. Über diesem liegend abwechselnd Textelemente mit Werten der Zahlenabstände sowie Griffpunkte.

In der Gruppe der „GrabDots“ existieren insgesamt zwölf vollflächige Kreise angeordnet in einer $3x4$ Matrix. Diese Punkte werden nach jedem Schrittwert in das Zahlenband eingefügt.

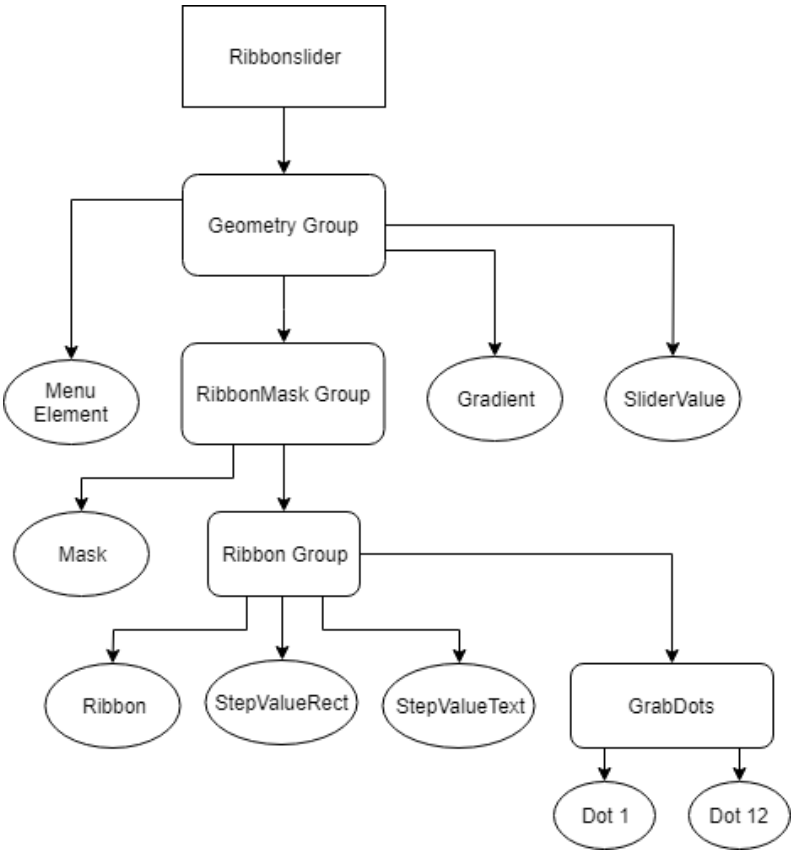


Abbildung 19: Paper.js Szenengraph des implementierten Ribbonslider. Ovale entsprechen grafischer Elemente, abgerundete Rechtecke repräsentieren Gruppen. Quelle: *Eigene Darstellung.*

Programmierung Der Ribbonslider erweitert die Klasse der Menüeinträge, diese entspricht einem normalen Menüeintrag mit beschreibendem Text und Icon. Menüeinträge erweitern die Klasse der Gruppe aus dem Paper.js Framework.

Die UML-Darstellung des Ribbonslider ist in Anhang B enthalten.

Der Schieberegler besitzt ein öffentliches Attribut „settings“, das die Konfigurationsdaten enthält. Die Konfiguration besteht aus fünf variablen Attributen. Die Angabe des Minimal- und Maximalwertes steuert den Wertebereich des Ribbonslider, der Initialwert definiert die Position auf dem Zahlenband nach der ersten Aktivierung. Die Angabe der Schrittgröße und Schrittdistanz beschreibt den Abstand zwischen zwei Werten sowie den Abstand in Pixeln zwischen zwei Schritten auf dem Zahlenband.

Die Länge des Zahlenbands ist abhängig der gewählten Einstellungen des Schiebereglers. Berechnet wird diese mit der Formel:

$$length = ((max - min)/stepSize) * stepDistance$$

Ein Schieberegler mit einem Wertebereich von -100 – 100 mit einer Schrittgröße von 1 und einer Schrittdistanz von $100px$ besitzt eine Länge von $20000px$.

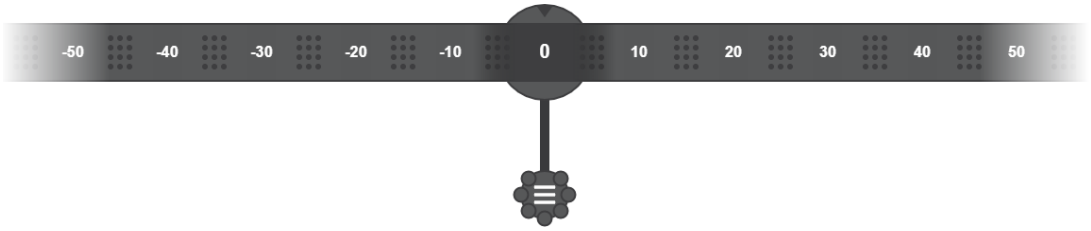
Der Ribbonslider implementiert für eine vereinfachte Interaktion die sogenannte „Pointer Lock API“ [6]. Diese Browser-API erlaubt den Zugriff auf die Bewegungsdaten des Eingabegeräts.

Der Vorteil des Pointer Locks ist, dass Bewegungen nicht auf die Größe des Bildschirms beschränkt sind, sondern ins unendliche fortgeführt werden können. Ohne Einsatz des Pointer Locks kann das Zahlenband des Schiebereglers mit der Maus nur bis zum Rand des Bildschirms bewegt werden, für eine nachfolgende Bewegung müsste die Maus gelöst und neu positioniert werden.

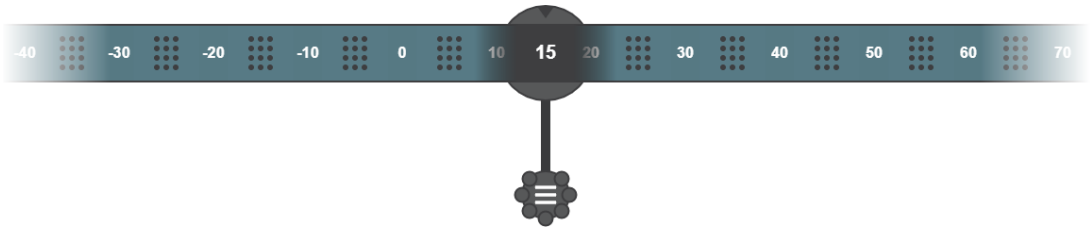
Mit dem Pointer Lock wird der Mauszeiger nach der Aktivierung an einem Punkt festgehalten, es werden statt absoluter Bewegungen ausschließlich die Differenz der Bewegung zu der vorherigen Position übermittelt. Somit ist die Größe der Bewegung nicht mehr auf die Größe des Bildschirms begrenzt und das Zahlenband kann „unendlich“ weit bewegt werden.

Abbildung 20a zeigt die Umsetzung des Ribbonslider in dem Marking-Menü nach getätigter Auswahl. Zu sehen ist der Wertebereich -50 bis 50 mit einer Schrittgröße von 10. In Abbildung 20b ist der Ribbonslider während einer aktiven Bedienung dargestellt. Der aktive Status wird durch eine bläuliche Einfärbung des Zahlenbands verdeutlicht.

Neben der Bewegung des Zahlenbandes können Werte durch Auswählen eingestellt werden. Abbildung 21a zeigt den Schieberegler vor der Auswahl eines Werts des Zahlenbands. Der Ribbonslider wird analog des ausgewählten Werts verschoben. Der finale Zustand ist in Abbildung 21b dargestellt.



(a) Darstellung des Ribbonslider nach Auswahl aus Marking-Menü.



(b) Darstellung des Ribbonslider während aktiver Bedienung.

Abbildung 20: Darstellung des implementierten Ribbonslider. Konfigurierter Wertebereich: -100 – 100, Schrittgröße 10, Schrittdistanz: $100px$. Quelle: *Eigener Screenshot.*

(a) Hervorhebung eines Werts vor Auswahl

(b) Ribbonslider nach Auswahl eines Werts des Zahlenbands

Abbildung 21: Direkte Auswahl eines Werts durch Klick auf dem Zahlenband. Quelle: *Eigener Screenshot.*

Abbildung 22: Ribbonslider nach Verringerung der Anzahl der Griffpunkte. Quelle: *Eigener Screenshot.*

4.5 Evaluation

Funktional erfüllt der Ribbonslider alle obligatorischen und wünschenswerten Anforderungen aus Tabelle 3. Fakultative Anforderungen wie die Nichtlinearität können durch den Schieberegler realisiert werden, eine Implementierung ist in einer zukünftigen Arbeit geplant.

Im Zuge einer informellen Evaluation wurde der Schieberegler vier Personen der Pilotstudie aus Kapitel 4.3.: *Ergebnisse der Papierprototypen* zur Anwendung vorgelegt.

Allen Teilnehmern war die Bedienung des Ribbonslider ohne Erklärung klar. Die Performance wurde als gut bewertet, dies ging aus dem Einsatz des Pointer Locks hervor, durch den die Extremwerte des Schiebereglers schnell erreicht werden konnten.

Ein Anwender kommentierte die Anzahl der Griffpunkte, diese suggerierten fehlende Werte zwischen zwei Wertschritten. Abbildung 22 zeigt eine Umsetzung des Ribbonslider mit verringerter Griffpunktanzahl. Die Griffpunkte stehen hier visuell nicht mehr hervor, zeigen aber weiterhin die Anfassbarkeit des Zahlenbandes an.

5 Zusammenfassung

Der Verlauf dieser Arbeit hat einen Einblick in die historische Entwicklung und den Stand der Technik von Pie-Menüs gegeben. Beschrieben wurde der Kontext aus dem Pie-Menüs entstanden sind und welche Gesetzmäßigkeit diese umsetzen, die die Bedienung performanter gegenüber linearen Menüstrukturen macht. Das Kapitel der Bedienarten betrachtet Beispiele einiger Einsatzgebiete und Interaktionsformen für Pie-Menüs. Besonders Computeranwendungen in denen Aktionen häufig ausgeführt werden, zeigten eine vermehrte Umsetzung von Pie-Menüs auf. Dies konnte auf den Geschwindigkeitsvorteil zurückgeführt werden, der sich in der Auswahl von Menüeinträgen ergibt.

Eine Beschreibung von Marking-Menüs, die Pie-Menüs um Gestennavigation erweitern, enthielt neben der historischen Entwicklung auch die Stärken und Schwächen dieser Menüs. Das Marking-Menü dient als Grundlage aller Implementierungen dieser Arbeit.

Der Stand der Technik untersuchte zusätzlich gängige Steuerelemente einer Benutzeroberfläche sowie die Umsetzung dieser Widgets in Pie-Menüs. Ein Großteil der standardisierten Steuerelemente kann trivial in einem Pie- bzw. Marking-Menü umgesetzt werden. Das Element des Schiebereglers ging aus der Untersuchung als nicht trivial in der Umsetzung hervor und wurde im Verlauf der Arbeit genauer betrachtet.

Die Analyse des Steuerelements des Schiebereglers umfasste neben der Beschreibung der Grundlagen, eine Auflistung der verschiedenen Funktionalitäten und Formen von Schiebereglern. Eine anschließende Beschreibung über die Integration des Steuerelements in Pie-Menüs zeigte zwei Umsetzungen und beschrieb deren Probleme.

Zuvor erfolgte die Implementierung eines Marking-Menü Frameworks. Beschrieben wurden die zu erfüllenden Anforderungen eines Marking-Menü Frameworks sowie die Umsetzung, Strukturierung und Interaktionsmöglichkeiten mit dem Menü.

Die eingangs gestellte Frage über die Integration von Schiebereglern in Marking-Menüs und die zu erfüllenden Anforderungen wird im Kapitel über Prototypen beantwortet. Obligatorische Anforderungen umfassen, neben der Integration in das Menü, die Anzeige des aktuell eingestellten Werts sowie das Beenden der Interaktion durch eine dedizierte Aktion.

Die Aufstellung der Anforderungen geschah unter Betrachtung verwandter Arbeiten und Analyse des Standardfunktionsumfangs des klassischen Schiebereglerelements. Zusammengefasst wurden gleichartige Anforderungen in den zwei Kategorien „funktional“ und „nichtfunktional“. In der anschließenden Gewichtung der aufgestellten Anforderungen erfolgte die Unterteilung in die Kategorien „obligatorisch“ bis „fakultativ“.

Anhand der aufgestellten und gewichteten Anforderungen, erfolgte die Erstellung von vier Prototypen, die in ihrer Bedienung zueinander orthogonal auftreten. Neben der Beschreibung der Funktionsweise und Anforderungsabdeckung jedes Schiebereglers erfolgte zusätzlich eine kurze Analyse in Hinblick auf die Bedienbarkeit durch Toucheingaben.

Interaktionen mit den Schiebereglern wurden mittels Papierprototypen visualisiert. Anhang A enthält die erstellten Schieberegler. In einer Pilotstudie wurden die Prototypen sowie dem FaST Slider aus den verwandten Arbeiten acht Probanden vorgelegt und nach den Kriterien Spaß, Performance und Verständlichkeit bewertet.

Das Ergebnis der Studie ließ erste Vermutungen über gut funktionierende Schieberegler in Marking-Menüs zu. Anhand abgegebener Nutzerkommentare erfolgte eine Überarbeitung des bestbewerteten Schiebereglers. Die überarbeitete Form des Ribbonslider aus Kapitel 4.2.2.: *Ribbonslider* wurde in das erstellte Marking-Menü Framework integriert.

Die Evaluation des Schiebereglers umfasste eine erneute Vorlage der Umsetzung vor vier Testpersonen. Keiner der Anwender benötigte eine Erklärung der Bedienung. Der Schieberegler wurde als performant und verständlich bewertet.

Der Ribbonslider deckt die aufgestellten Anforderungen in Gänze ab und war durch die Testpersonen problemlos als Schieberegler zu erkennen und zu benutzen. Die Integration des Schiebereglers in die Menüstruktur des Marking-Menüs kann als gelungen bewertet werden. Eine reine Kombination des Schiebereglerwidgets mit der Struktur des Marking-Menüs wurde vermieden. Die Bedienvorteile der Gestennavigation konnten in den Schieberegler integriert werden.

5.1 Ausblick

Das entwickelte Marking-Menü Framework zeigt beispielhaft, wie das Steuerelement des Schiebereglers integriert werden kann. Das Marking-Menü deckt mit der Umsetzung der restlichen trivialen Widgets einen Großteil des Funktionsumfangs heutiger Benutzeroberflächen ab.

In zukünftigen Arbeiten ist vorgesehen das Marking-Menü Framework eingehend zu analysieren und zu erweitern. Insbesondere soll die Performance zwischen Touch- und Mauseingaben in einem solchen Menü untersucht werden. Vermutlich kann das Zeichnen von Gesten mit der Hand performanter sein als mit der Maus.

Die Pilotstudie der Papierprototypen soll für eine fundierte Aussage erweitert werden und mit Umsetzungen der Schieberegler in dem Marking-Menü Framework getestet werden. Betrachtet werden soll neben dem Vergleich der Schieberegler untereinander zusätzlich die Bedienung mit Toucheingaben.

Die Welt der Pie- und Marking-Menü Frameworks wurde um eine moderne und großflächig einsetzbare Lösung erweitert.

> Die tiefgreifendsten Technologien sind diejenigen, die verschwinden. Sie verweben sich in das Gewebe des Alltags, bis sie von ihm nicht mehr zu unterscheiden sind.
>
> — Frei nach Mark Weiser

Literaturverzeichnis

[1] Christopher Ahlberg und Ben Shneiderman. „The alphaslider“. In: *Proceedings of the SIGCHI conference on Human factors in computing systems celebrating interdependence - CHI '94*. the SIGCHI conference (Boston, Massachusetts, United States, 24. Apr. 1994). Hrsg. von Beth Adelson, Susan Dumais und Judith S. Olson. New York, New York, USA: ACM Press, 1994, S. 365–371. ISBN: 0897916506. DOI: 10.1145/191666.191790.

[3] Jack Callahan u. a. „An empirical comparison of pie vs. linear menus“. In: *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '88*. the SIGCHI conference (Washington, D.C., United States, 15. Mai 1988). Hrsg. von J. J. O'Hare. New York, New York, USA: ACM Press, 1988, S. 95–100. ISBN: 0201142376. DOI: 10.1145/57167.57182.

[7] Ronald Ecker u. a. „pieTouch“. In: *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services - MobileHCI '09*. the 11th International Conference (Bonn, Germany, 15. Sep. 2009). Hrsg. von Reinhard Oppermann u. a. New York, New York, USA: ACM Press, 2009, S. 1. ISBN: 9781605582818. DOI: 10.1145/1613858.1613887.

[9] Paul M. Fitts. „The information capacity of the human motor system in controlling the amplitude of movement“. In: *Journal of Experimental Psychology* 47.6 (1954), S. 381–391. ISSN: 0022-1015. DOI: 10.1037/h0055392.

[11] Wilbert O. Galitz. *The essential guide to user interface design. An introduction to GUI design principles and techniques*. 3. ed. Indianapolis, Ind.: Wiley, 2007. 1 online resource (XXVII, 857 Seiten). ISBN: 9780470053423.

[12] Jesse J. Garrett. *The elements of user experience. User-centered design for the Web and beyond*. 2nd ed. Voices that matter. Berkeley, CA: New Riders, 2011. Online-Ressource (1 online resource xviii, 172. ISBN: 9780321683687.

[13] Sascha Gebhardt u. a. „Extended pie menus for immersive virtual environments“. eng. In: *IEEE transactions on visualization and computer graphics* 19.4 (2013). Journal Article Research Support, Non-U.S. Gov't, S. 644–651. DOI: 10.1109/TVCG.2013.31. eprint: 23428449.

[14] François Guimbretiére und Terry Winograd. „FlowMenu“. In: *Proceedings of the 13th annual ACM symposium on User interface software and technology - UIST '00*. the 13th annual ACM symposium (San Diego, California, United States, 6. Nov. 2000). Hrsg. von Mark Ackerman und Keith Edwards. New York, New York, USA: ACM Press, 2000, S. 213–216. ISBN: 1581132123. DOI: 10.1145/354401.354778.

[16] Don Hopkins. „The design and implementation of pie menus“. In: *Dr. Dobb's Journal* 16.12 (1991), S. 16–26.

[18] Alexander Kulik u. a. „The Pie Slider: Combining Advantages of the Real and the Virtual Space“. In: *Smart Graphics*. Hrsg. von Andreas Butz u. a. Bd. 5531. Lecture Notes in Computer Science. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, S. 93–104. ISBN: 978-3-642-02114-5. DOI: 10.1007/978-3-642-02115-2_8.

[19] Gordon P. Kurtenbach. „The Design and Evaluation of Marking Menus“. UMI Order No. GAXNN-82896. Toronto, Ont., Canada, Canada: University of Toronto, 1993.

[20] Gordon P. Kurtenbach und William A. S. Buxton. „The limits of expert performance using hierarchic marking menus“. In: *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '93*. the SIGCHI conference (Amsterdam, The Netherlands, 24. Apr. 1993). Hrsg. von Bert Arnold, Gerrit van der Veer und Ted White. New York, New York, USA: ACM Press, 1993, S. 482–487. ISBN: 0897915755. DOI: 10.1145/169059.169426.

[21] Gordon P. Kurtenbach, Abigail Sellen und William A. S. Buxton. „An Empirical Evaluation of Some Articulatory and Cognitive Aspects of Marking Menus“. In: *Human–Computer Interaction* 8.1 (1993). PII: 784766234, S. 1–23. ISSN: 0737-0024. DOI: 10.1207/s15327051hci0801_1.

[23] Ian S. MacKenzie. „A note on the information-theoretic basis of Fitts' law“. eng. In: *Journal of motor behavior* 21.3 (1989). Journal Article, S. 323–330. ISSN: 0022-2895. DOI: 10.1080/00222895.1989.10735486. eprint: 15136269.

[24] Ian S. MacKenzie. „Fitts' Law“. In: *The Wiley handbook of human computer interaction set*. Hrsg. von Kent L. Norman. First edition. Hoboken, NJ: John Wiley & Sons, 2018, S. 349–370. ISBN: 978-1-118-97727-9.

[25] Ian S. MacKenzie. „Fitts' Law as a Research and Design Tool in Human-Computer Interaction“. In: *Human–Computer Interaction* 7.1 (1992), S. 91–139. ISSN: 0737-0024. DOI: 10.1207/s15327051hci0701_3.

[26] Ian S. MacKenzie. „Movement Time Prediction in Human-Computer Interfaces“. In: *Readings in Human–Computer Interaction*. Elsevier, 1995, S. 483–493. ISBN: 9780080515748. DOI: 10.1016/b978-0-08-051574-8.50050-9.

[27] Michael McGuffin, Nicolas Burtnyk und Gordon P. Kurtenbach. „FaST Sliders: Integrating Marking Menus and the Adjustment of Continuous Values“. en-ca. In: *Graphics Interface 2002. Proceedings ; Calgary, Alberta, 27 - 29 May 2002, [Canadian Human-Computer Communications Society*. Hrsg. von Wolfgang Stürzlinger. Mississauga: Canadian Information Processing Society, 2002, S. 35–42. ISBN: 1-56881-183-7. DOI: 10.20380/GI2002.05.

[30] Krystian Samp und Stefan Decker. „Supporting menu design with radial layouts“. In: *Proceedings of the International Conference on Advanced Visual Interfaces - AVI '10*. the International Conference (Roma, Italy, 26. Mai 2010). Hrsg. von Giuseppe Santucci. New

York, New York, USA: ACM Press, 2010, S. 155. ISBN: 9781450300766. DOI: 10.1145/1842993.1843021.

[31] Farzan Sasangohar, Ian S. MacKenzie und Stacey D. Scott. „Evaluation of Mouse and Touch Input for a Tabletop Display Using Fitts' Reciprocal Tapping Task". In: *Human Factors and Ergonomics Society Annual Meeting Proceedings* 53.12 (2009), S. 839–843. ISSN: 10711813. DOI: 10.1518/107118109x12524442637787.

[34] Mark A. Tapia und Gordon P. Kurtenbach. „Some design refinements and principles on the appearance and behavior of marking menus". In: *Proceedings of the 8th annual ACM symposium on User interface and software technology - UIST '95*. the 8th annual ACM symposium (Pittsburgh, Pennsylvania, United States, 15. Nov. 1995). Hrsg. von George Robertson. New York, New York, USA: ACM Press, 1995, S. 189–195. ISBN: 089791709X. DOI: 10.1145/215585.215973.

[35] Pawan Vora. *Web application design patterns*. eng. Interactive technologies. Amsterdam: Morgan Kaufmann/Elsevier, 2009. 429 S. ISBN: 9780123742650. URL: http://site.ebrary.com/lib/alltitles/docDetail.action?docID=10286099.

[36] Neil E. Wiseman, H. U. Lemke und J. O. Hiles. „PIXIE: A new approach to graphical man-machine communication". In: *Proceedings of 1969 CAD Conference Southhampton*. Bd. 463. 1969.

Onlinereferenzen

[2] Blender Documentation Team. *Blender 2.79 Manual. User Interface - Pie Menus*. Hrsg. von Blender Documentation Team. 2017. URL: https://docs.blender.org/manual/en/latest/interface/controls/buttons/menus.html#pie-menus (besucht am 25. 07. 2019).

[4] Capcom. *MONSTER HUNTER: WORLD Official Web Manual. Using the Custom Radial Menu*. Hrsg. von Capcom. 2018. URL: https://game.capcom.com/manual/MHW/en/ps4/page/5/6 (besucht am 26. 07. 2019).

[5] Apurva Chakke. *Expense Input Interaction*. Hrsg. von Dribble. 2015. URL: https://dribbble.com/shots/2219579-Expense-Input-Interaction (besucht am 01. 08. 2019).

[6] ebidel. *Pointer Lock API*. Hrsg. von MDN web docs. 2011. URL: https://developer.mozilla.org/en-US/docs/Web/API/Pointer_Lock_API$revision/1528001 (besucht am 05. 09. 2019).

[8] Julian Eisel und Psy-Fi. *Blender 2.72: User Interface. Pie menus*. Hrsg. von Blender Documentation Team. 2016. URL: https://archive.blender.org/wiki/index.php/Dev:Ref/Release_Notes/2.72/UI/ (besucht am 25. 07. 2019).

[10] Fuji Photo Film (Europe) G.m.b.H. *FinePix S602Zoom. Ownser's Manual*. 2002. URL: https://www.fujifilmusa.com/shared/bin/FX-S602_Manual.pdf (besucht am 27. 08. 2019).

[15] Aurora Harley. *Slider Design: Rules of Thumb*. Hrsg. von Nielsen Norman Group. 2015. URL: https://www.nngroup.com/articles/gui-slider-controls/ (besucht am 31. 07. 2019).

[17] Don Hopkins u. a. *Theta Menus Proposal and Pie Menu Designs*. Hrsg. von Don Hopkins. 1986. URL: https://web.archive.org/web/20110611185226/http://www.donhopkins.com/drupal/node/82# (besucht am 23. 07. 2019).

[22] Jürg Lehni und Jonathan Puckey. *Paper.js. Features*. 2011. URL: http://paperjs.org/features/ (besucht am 06. 08. 2019).

[28] Octfx. *Visualize an object as non movable*. Hrsg. von Stack Exchange. User Experience. 2019. URL: https://ux.stackexchange.com/questions/127714/visualize-an-object-as-non-movable (besucht am 26. 08. 2019).

[29] RWTH Aachen. *ViSTA Virtual Reality Toolkit*. Hrsg. von RWTH Aachen. 2008. URL: http://www.itc.rwth-aachen.de/cms/IT-Center/Forschung-Projekte/Virtuelle-Realitaet/Infrastruktur/~fgmo/ViSTA-Virtual-Reality-Toolkit/ (besucht am 26. 07. 2019).

[32] Simon Schneegans. *Gnome-Pie. A pie menu launcher for linux*. Hrsg. von Simon Schneegans. 2011. URL: https://github.com/Simmesimme/Gnome-Pie (besucht am 25. 07. 2019).

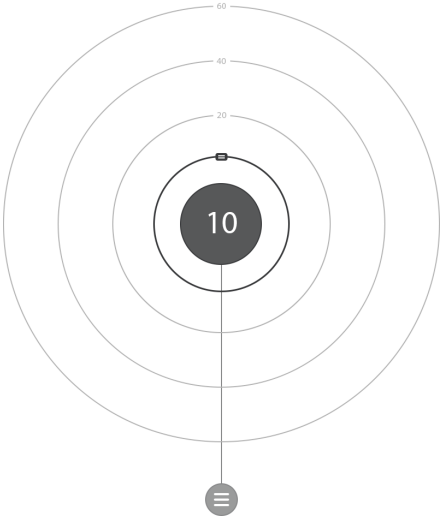
[33] Simon Schneegans. *Gnome-Pie*. 2011. URL: https://wiki.ubuntuusers.de/GNOME-Pie/a/revision/969753/ (besucht am 30. 07. 2019).

A Papierprototypen

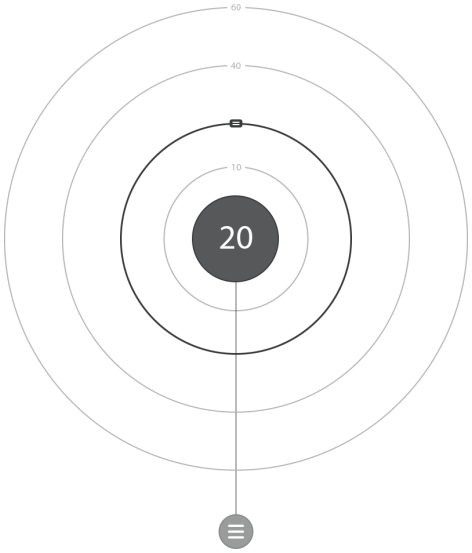
Papierprototypen

A.1 Circleslider

(a) Startzustand



(b) Zustand nach Aktivierung



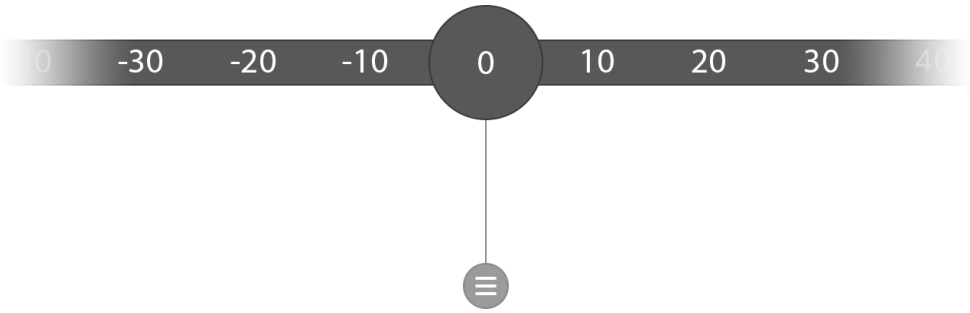
(c) Zustand nach Wertänderung

(d) Endzustand

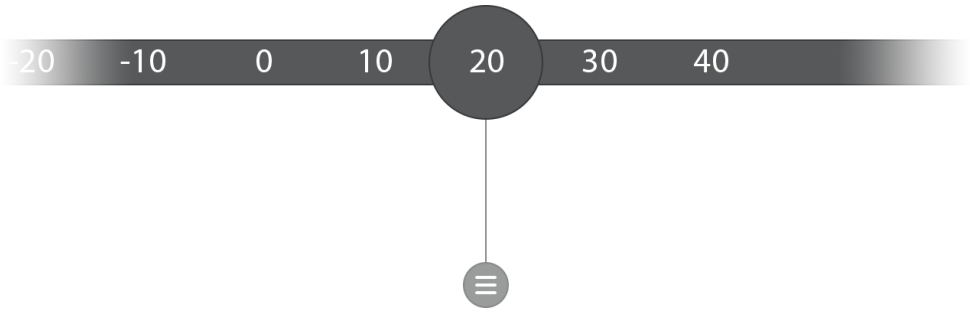
Prototyp A.1: Reihenfolge der vorgelegten Papierprototypen des Circlesliders. Mit der Vorlage des Startzustands (a) wurden die Probanden angewiesen den nördlichsten Eintrag des Menüs auszuwählen. Nach der Auswahl erfolgte die Vorlage des Zustands (b)mit der Aufgabe den Wert des Schiebereglers auf 20 zu verändern. Der nachfolgende Zustand (c) wurde nach der Einstellung des Werts vorgelegt. Der Testgang endete mit dem Menüzustand (d).

A.2 Ribbonslider

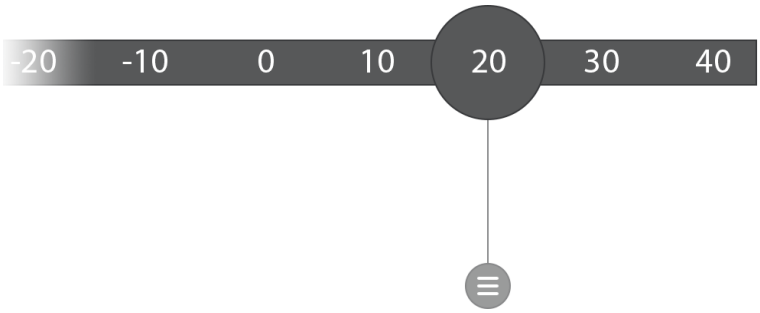
(a) Startzustand



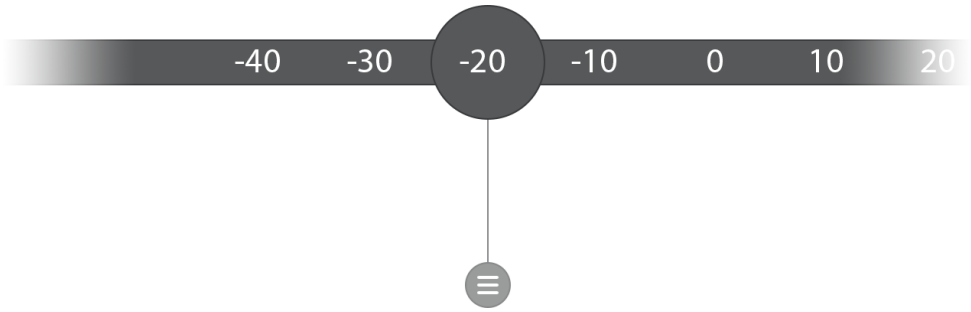
(b) Zustand nach Aktivierung



(c) Zustand nach Wertänderung



(d) Zustand (c) alternative Darstellung



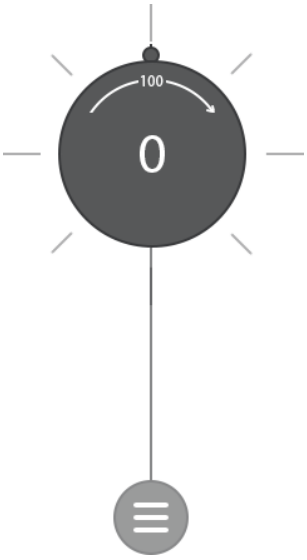
(e) Zustand nach negativer Wertänderung

(f) Endzustand

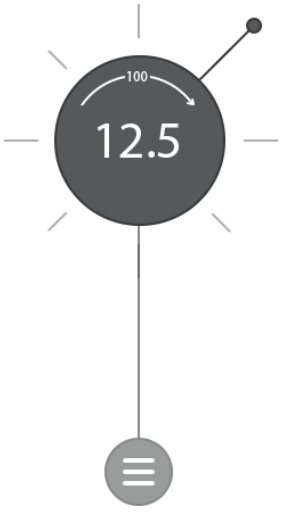
Prototyp A.2: Reihenfolge der vorgelegten Papierprototypen des Ribbonsliders. Mit der Vorlage des Startzustands (a) wurden die Probanden angewiesen den nördlichsten Eintrag des Menüs auszuwählen. Nach der Auswahl erfolgte die Vorlage des Zustands (b) mit der Aufgabe den Wert des Schiebereglers auf 20 zu verändern. Der nachfolgende Zustand (c) und (d) wurde nach der Einstellung des Werts vorgelegt. Probanden wurden nach der Präferenz des dargestellten Endzustand des Wertebereichs befragt. Abschließend wurde die Aufgabe gestellt, die Wertigkeit auf -20 zu verändern (e). Der Testgang endete mit dem Menüzustand (f).

A.3 Crankslider

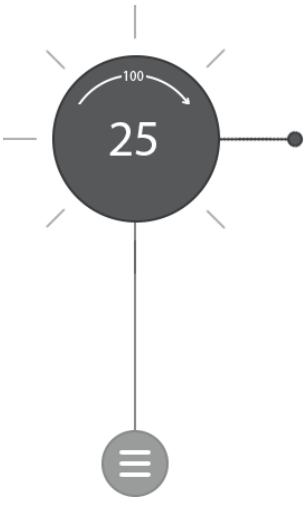
(a) Startzustand



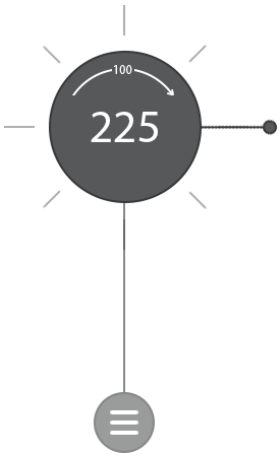
(b) Zustand nach Aktivierung



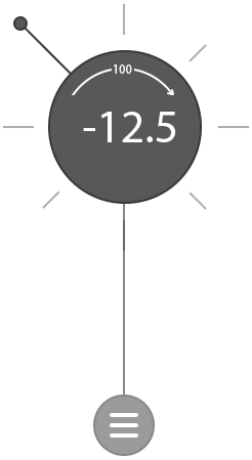
(c) Zustand während Wertänderung



(d) Zustand nach Wertänderung



(e) Zustand nach Wertänderung



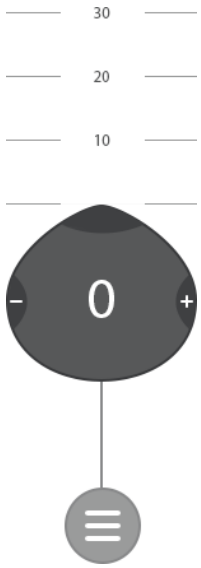
(f) Zustand nach negativer Wertänderung

(g) Endzustand

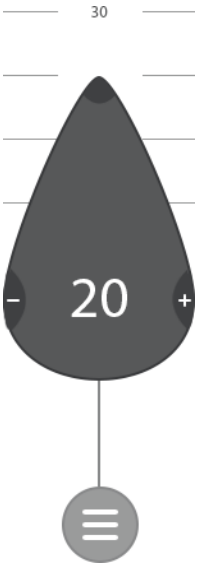
Prototyp A.3: Reihenfolge der vorgelegten Papierprototypen des Cranksliders. Mit der Vorlage des Startzustands (a) wurden die Probanden angewiesen den nördlichsten Eintrag des Menüs auszuwählen. Nach der Auswahl erfolgte die Vorlage des Zustands (b)mit der Aufgabe den Wert des Schiebereglers auf 25 zu verändern. Während der Drehung des Schiebereglers im Uhrzeigersinn wurde erst der Zustand (c), dann der Zustand (d) vorgelegt. Bei einer Linksdrehung der Schiebereglers wurde der Zustand (f) vorgelegt. Nach erfolgreicher Einstellung des Werts 25 wurde der Zustand des Schiebereglers auf (b) zurückgesetzt. Der abschließende Zustand (e) wurde, nach der Aufgabe den Schiebereglerwert auf 225 zu stellen, vorgelegt. Der Testgang endete mit dem Menüzustand (g).

A.4 Morphslider

(a) Startzustand



(b) Zustand nach Aktivierung



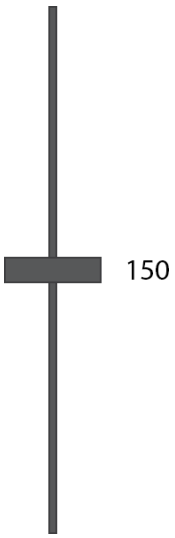
(c) Zustand nach Wertänderung

(d) Endzustand

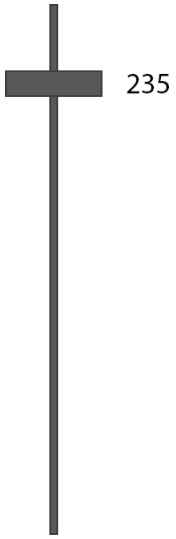
Prototyp A.4: Reihenfolge der vorgelegten Papierprototypen des Morphsliders. Mit der Vorlage des Startzustands (a) wurden die Probanden angewiesen den nördlichsten Eintrag des Menüs auszuwählen. Nach der Auswahl erfolgte die Vorlage des Zustands (b)mit der Aufgabe den Wert des Schiebereglers auf 20 zu verändern. Der nachfolgende Zustand (c) wurde nach der Einstellung des Werts vorgelegt. Der Testgang endete mit dem Menüzustand (d).

A.5 FaST Slider

(a) Startzustand



(b) Zustand nach Aktivierung



(c) Zustand nach Wertänderung

(d) Endzustand

Prototyp A.5: Reihenfolge der vorgelegten Papierprototypen des FaST Sliders, die Darstellung entsprechen dem von McGuffin, Burtnyk und Kurtenbach [27] beschriebenen Schieberegler mit farblichen Anpassungen. Mit der Vorlage des Startzustands (a) wurden die Probanden angewiesen den nördlichsten Eintrag des Menüs auszuwählen. Nach der Auswahl erfolgte die Vorlage des Zustands (b)mit der Aufgabe den Wert des Schiebereglers auf 235 zu verändern. Die Probanden wurde darauf hingewiesen, dass der angezeigte Wert des Startzustandes (a) von dem Folgezustand (b) abweicht. Der nachfolgende Zustand (c) wurde nach der Einstellung des Werts vorgelegt. Der Testgang endete mit dem Menüzustand (d).

B Ribbonslider UML