Institut für Visualisierung und Interaktive Systeme

Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Bachelorarbeit Nr. 212

Gruppierung von Eye-Tracking-Daten mittels geeigneter Ähnlichkeitsfunktionen

Frank Heyen

Studiengang:	Informatik
Prüfer/in:	Prof. Dr. Thomas Ertl
Betreuer/in:	Dipl.-Inf. Dominik Herr, Dipl.-Inf. Tanja Blascheck
Beginn am:	22. April 2015
Beendet am:	22. Oktober 2015
CR-Nummer:	H.1.2, H.5.2, J.4

Kurzfassung

Eye-Tracking gewann als Hilfsmittel zur Evaluation von Benutzerschnittstellen und Visualisierungen in den letzten Jahren stets an Beliebtheit. Ein Vergleich der Lösungsstrategien verschiedener Personen kann anhand der Blickpfade, auch Scanpaths genannt, durchgeführt werden. Für diese Aufgabe fehlt zurzeit noch eine optimale Methode. Bereits existierende Arbeiten verwenden unter anderem Algorithmen zum String-Vergleich, um die Ähnlichkeit zwischen Scanpaths zu ermitteln. Diese Algorithmen können durch Parameter beeinflusst werden. Auch eine Vorverarbeitung der Blickpfade ist durch Methoden mit weiteren Parametern möglich. Angesichts der Vielzahl von denkbaren Kombinationen ist eine Auswahl der optimalen Parameter schwer. In dieser Arbeit werden unterschiedliche Ansätze für den Vergleich von Scanpaths untersucht. Dazu gehören unter anderem die Levenshtein-Distanz und der Algorithmus von Needleman und Wunsch, die einen Wert für die Ähnlichkeit von Strings berechnen. Für diese Ansätze werden Erweiterungen zur Vorverarbeitung der Scanpaths und Einbeziehung weiterer Informationen in den Vergleich erarbeitet. Eine Evaluation in drei Versuchen mit generierten und real aufgezeichneten Eye-Tracking-Daten zeigt anschließend, welche der Parameterkonfigurationen sich in der Praxis bewähren.

Abstract

Eye tracking has recently become a popular technique for evaluating user interfaces and visualizations. A comparison of strategies used by participants to solve a task can be done using so-called scanpaths. There is still a need for an optimal method for the comparison of those paths. Previous approaches often use string comparison algorithms to determine the similarity between scanpaths. Various parameters can affect the behavoiur of those algorithms. Additionally, methods requiring even more parameters can pre-process scanpaths. Due to the number of possible combinations, choosing optimal parameters is a non-trivial task. Different existing approaches dealing with scanpath comparison are examined in this work, including the string comparison algorithms from Levenshtein and Needleman-Wunsch. Possible extensions regarding pre-processing and the inclusion of further information into the comparison are developed. The results are then evaluated in three experiments using generated and real world eye tracking data to demonstrate the performance of different parameter configurations in practical data analysis.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Verzeichnis der Listings

1 Einleitung

Eye-Tracking ist zu einer beliebten Technik in Industrie und Forschung geworden. Die Erfassung und Analyse der Augenbewegungen von Studienteilnehmern ermöglicht unter anderem Rückschlüsse auf die von ihnen verwendete Strategie bei der Lösung einer Aufgabe. Untersucht werden kann dabei beispielsweise, welche Objekte wann, wie lange und in welcher Reihenfolge angeschaut wurden. Eine länger andauernde Betrachtung eines Objektes kann sowohl Interesse als auch Probleme beim Verständnis widerspiegeln.

Aufschlussreich ist in vielen Fällen der Vergleich von mehreren Probanden bezüglich ihrer Lösungsstrategie. Statt einer bloßen Gegenüberstellung von statistischen Daten zur Verteilung der visuellen Aufmerksamkeit über Zeit und Raum, können auch sogenannte Scanpaths verglichen werden (siehe Abb. 1.1). Diese beschreiben den Pfad, den der Blick einer Person auf einem betrachteten Stimulus genommen hat, Blickpunkt für Blickpunkt. Es wurden bereits einige Techniken zum Vergleich dieser Pfade entwickelt, jedoch besteht weiterhin Bedarf nach einer Methode zur optimalen Ermittlung der Ähnlichkeit zwischen Scanpaths und damit der Strategie zweier Personen. Keiner der existierenden Ansätze kann in jeder Situation überzeugen.

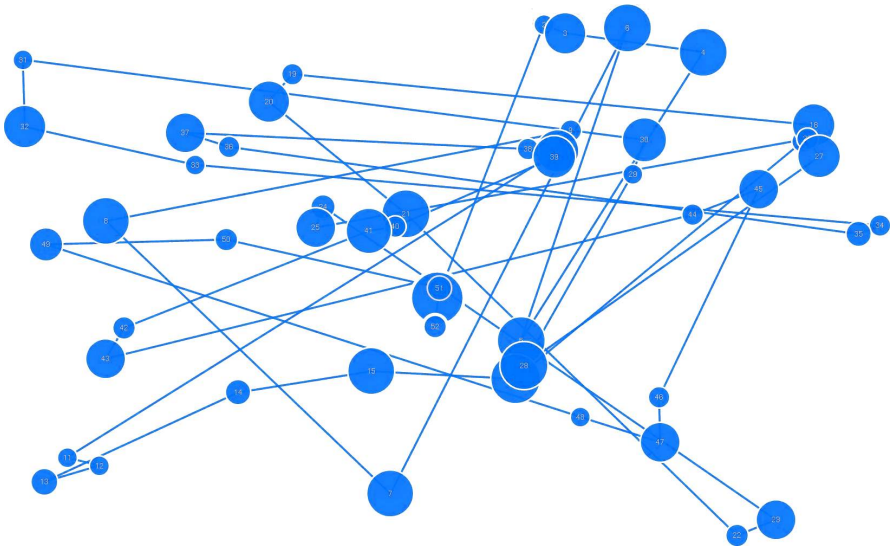


Abbildung 1.1: Beispiel eines Scanpaths, dargestellt durch Kreise für Fixationen (Blickpunkte) und Linien für Sakkaden (Augenbewegungen). Die Größe der Kreise visualisiert die Fixationsdauer.

Ziel dieser Arbeit ist eine Gegenüberstellung der vorhandenen Vergleichsmetriken sowie der Möglichkeiten, diese anzupassen oder zu erweitern. Dabei kommen unterschiedliche, teils datenabhängige Parameter zum Einsatz. In einer Evaluation soll herausgefunden werden, welche Metrik unter welcher Konfiguration am besten für den Vergleich von Scanpaths geeignet ist.

Diese Ausarbeitung ist in acht Kapitel gegliedert. Kapitel 2 und 3 behandeln die zum Verständnis benötigten Grundlagen sowie die Aufgabenstellung und den groben Lösungsansatz. Danach werden in Kapitel 4 bereits vorhandene, thematisch verwandte Arbeiten vorgestellt. Im fünften Kapitel wird das für diese Arbeit entwickelte Konzept erläutert. Die beiden folgenden Kapitel beschäftigen sich mit der Implementierung und Evaluation eines Prototypen. Zum Schluss werden die dabei entstandenen Ergebnisse im letzten Kapitel zusammengefasst und ein Ausblick auf mögliche weitere Forschungsarbeiten gezeigt.

2 Grundlagen

Dieses Kapitel behandelt die grundlegenden Themen dieser Arbeit, die für das Verständnis der darauffolgenden Kapitel relevant sind. Zunächst wird die Funktion des Auges und seiner Bewegungen erläutert. Im zweiten Abschnitt folgen einige Grundlagen zum *Eye-Tracking*. Daraufhin werden Ansätze zur Bestimmung der Ähnlichkeit von Scanpaths oder Zeichenketten vorgestellt. Abschließend wird auf die Themen *Clustering* und *Visualisierung* eingegangen.

2.1 Das menschliche Auge

Die optische Wahrnehmung ist der Primärsinn des Menschen mit einem Anteil von 95 Prozent an den von allen Sinnesorganen gesammelten Informationen [52]. In den folgenden Abschnitten werden der Aufbau, die Funktionsweise und die verschiedenen Bewegungstypen des menschlichen Auges erläutert.

2.1.1 Anatomie und Funktionsweise

Das menschliche Auge besteht im Wesentlichen aus dem von Muskeln bewegten Augapfel, der wiederum Pupille, Glaskörper und Netzhaut, auch Retina genannt, enthält. Eine Darstellung dieser anatomischen Bestandteile ist in Abb. 2.1 dargestellt. Um ein Objekt sehen zu können, muss dieses Licht abstrahlen oder reflektieren. Die vom Auge wahrgenommenen Lichtstrahlen gehen dann vom Objekt aus durch die Öffnung der Pupille. Um das Objekt scharfzustellen, wird die Linse von den sie umgebenden Muskeln so verformt, dass sie die Strahlen auf einen Bereich auf der Netzhaut projiziert. Die Netzhaut enthält zwei Arten von lichtempfindlichen Sinneszellen, die sogenannten *Stäbchen* und *Zapfen*. Stäbchen können schwaches Licht wahrnehmen, wodurch sie sich, im Gegensatz zu den Zapfen, zum Sehen bei Dunkelheit eignen. Mit ihnen kann jedoch nur ein monochromes Graustufenbild der Umgebung wahrgenommen werden. Bei ausreichendem Licht verbessert sich die Wahrnehmung daher stark durch die von den Zapfen aufgenommenen Farbinformationen. Es gibt drei Varianten von Zapfen, die für jeweils unterschiedliche Lichtspektren empfindlich sind. Dabei entsprechen die Zapfentypen S, M und L in etwa jeweils den Farben Blau, Grün und Rot. Der größte Teil der Netzhaut setzt sich fast ausschließlich aus Stäbchen und nur vereinzelt aus Zapfen zusammen, wodurch hier kein Farbsehen möglich ist. Hierhin wird durch die Linse das sogenannte *periphere Blickfeld* projiziert. Dies ist der Bereich des gesehenen Bildes, der nur unscharf wahrgenommen wird. Er macht den Hauptteil des Sichtbereiches aus. Scharfes Sehen findet hingegen nur in einem kleinen Bereich der Netzhaut statt. Er wird als Fovea Centralis oder gelber Fleck bezeichnet. In einem Umkreis von rund 1,5 Millimeter befinden sich hier keine Stäbchen, dafür etwa 140.000 Zapfen pro Quadratmillimeter [37, 44]. Durch diese hohe Dichte an Sinneszellen ergibt sich eine entsprechend

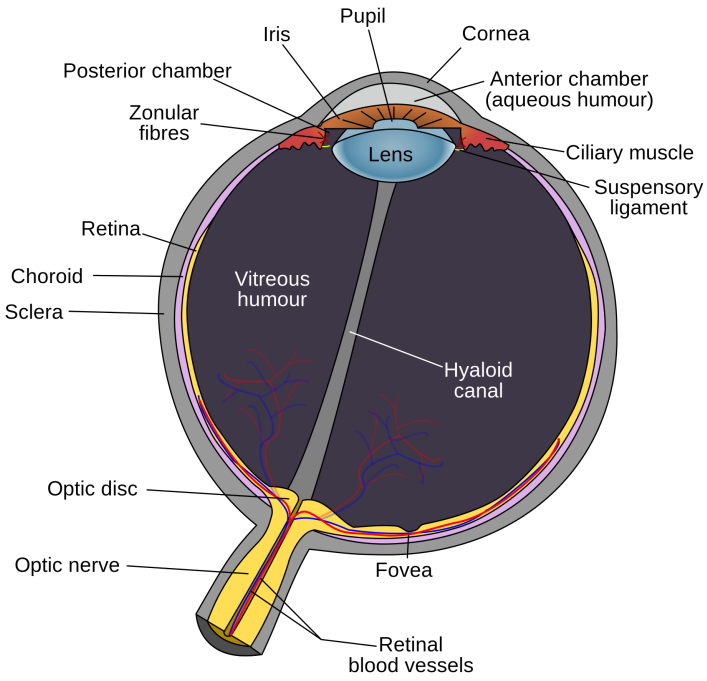


Abbildung 2.1: Die Anatomie des menschlichen Auges. Hier ist ein horizontaler Schnitt durch den rechten Augapfel abgebildet, wobei die Blickrichtung nach oben zeigt. Hinter Hornhaut (Cornea) und Pupille befindet sich die Linse, die das eintreffende Licht auf die Netzhaut (Retina) projiziert. Diese enthält lichtempfindliche Sinneszellen. Der Bereich der höchsten Auflösung befindet sich in der Fovea. An der Stelle, an der der Sehnerv die Netzhaut durchläuft, wird hingegen kein Licht wahrgenommen [47].

hohe Bildauflösung. Wie relevant dieser Bereich für die Wahrnehmung ist, zeigt sich auch darin, dass ungefähr die Hälfte der Nervenfasern des Sehnervs ausschließlich die dort aufgenommenen Informationen weiterleiten [16]. Um ein größeres Bild scharf wahrnehmen zu können, muss jedes markante Detail, vor allem Punkte und Linien, vom Auge so fokussiert werden, dass das von ihm ausgehende Licht in die Fovea gelangt. Um dies zu erreichen, wird der komplette Augapfel durch drei Muskelpaare um Nick-, Gier- und Rollachse gedreht [15]. Dieser Bewegungsapparat ist in Abb. 2.2 dargestellt.

2.1.2 Metriken zur Augenbewegung

Durch die Bewegungen, die das Auge zum Fokussieren verschiedener Punkte im Raum vornimmt, ergeben sich verschiedene Ereignisse. Diese werden im Folgenden aufgeführt. Ein Vergleich der typischerweise vorkommenden Werte für Dauer, Amplitude und Geschwindigkeit ist in der Tabelle 2.1 zu finden.

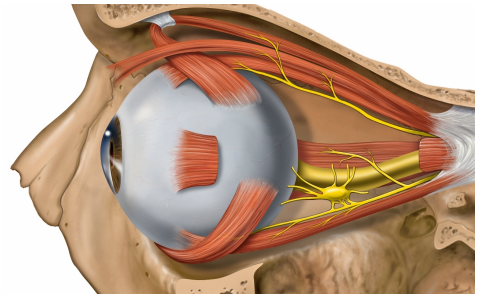


Abbildung 2.2: Der Bewegungsapparat des Auges. Drei Muskelpaare steuern die Bewegung des Augapfels, um schnell und präzise einen Blickpunkt nach dem anderen zu fokussieren. Vom Augapfel aus nach rechts verlaufen die Muskeln zur Neigung der Blickrichtung nach oben und unten. Der Muskel, der in dieser Graphik vor dem Augapfel zu sehen ist und sein Konterpart auf der gegenüberliegenden Seite neigen den Blick seitwärts. Die verbleibenden, diagonalen Muskeln rollen den Augapfel um die Blickachse herum. Das Bild wurde übernommen von P. J. Lynch [25] und zugeschnitten.

Sakkade. Als Sakkade wird der Sprung von einem Fokuspunkt zum nächsten bezeichnet. Sie ist die schnellste Bewegung, die der Mensch ausführen kann. Ihre Dauer beträgt etwa 30 bis 80 Millisekunden. Um den angezielten Punkt zu fokussieren, sind oft mehrere aufeinanderfolgende Sakkaden notwendig. Ein Grund dafür ist die Tatsache, dass selten der gerade Weg genommen wird und sich während einer durchgeführten Sakkade das Zielobjekt relativ zum Kopf des Betrachters bewegt haben kann. Weiterhin kommt es oft vor, dass über das Ziel hinaus geschossen wird, beispielsweise bei Sakkaden über große Abstände. Dieser Fehler wird dann durch als *Glissaden* bezeichnete Korrektur-Sakkaden berichtigt. Während eine Sakkade stattfindet, ist die Wahrnehmung unterbrochen, sodass der Mensch für diesen kurzen Moment blind ist.

Fixation. Eines der wichtigsten Ereignisse ist die Fixation. Diese etwas irreführende Bezeichnung steht für einen Moment, in dem das Auge zwar nicht komplett stillsteht, aber immerhin innerhalb eines gewissen Bereichs *verweilt*. Fixationen können kurz sein, etwa in der Größenordnung von 10 Millisekunden, sich aber auch über eine Zeitspanne von mehreren Sekunden erstrecken [15]. In der Regel findet während einer Fixation die Aufnahme von visueller Information durch das Auge in das Gehirn statt.

Drift. Drifts finden ebenfalls während Fixationen statt. Sie bezeichnen ein langsames Abgleiten des Auges von seinem eigentlich anvisierten Ziel.

Mikrosakkade. Um nicht zu weit von der ursprünglich fixierten Stelle abzudriften, korrigiert der Bewegungsapparat des Auges die durch Drifts entstandenen Abweichungen mittels Mikrosakkaden.

Verfolgung (Smooth Pursuit). Folgt das Auge einem sich eher langsam bewegenden Objekt, geschieht dies durch einen sogenannten Smooth Pursuit. Im Gegensatz zu Sakkaden ist hier eine Bewegung des fixierten Objekts eine notwendige Voraussetzung.

Typ	Dauer (ms)	Amplitude	Geschwindigkeit
Fixation	200-300	-	-
Sakkade	30-80	4-20°	30-500°/s
Glissade	10-40	0.5-2°	20-140°/s
Smooth Pursuit	-	-	10-30°/s
Mikrosakkade	10-30	10-40′	15-50°/s
Beben	-	<1′	20′/s (Spitze)
Drift	200-1000	1-60′	6-25′/s

Tabelle 2.1: Vergleich von typischen Werten der verschiedenen Metriken für Augenbewegungen. Räumliche Daten sind hier in visuellen Grad (°) oder Minuten (′, 60′=1°) angegeben. Die Daten wurden übernommen von Holmqvist, Nyström et al. [15].

2.2 Eye-Tracking

Eye-Tracking, zu deutsch in etwa *Blickerfassung*, bezeichnet die Aufnahme von Augenpositionen und -bewegungen. In diesem Abschnitt sollen der Nutzen und die für die Durchführung von Studien verwendeten Techniken aufgezeigt werden.

2.2.1 Motivation und Anwendungsmöglichkeiten

Es wird bei der Durchführung von Eye-Tracking-Studien im Allgemeinen davon ausgegangen, dass die fixierte Stelle eines Bildes auch derjenigen Information entspricht, mit dem das Gehirn des Probanden gerade beschäftigt ist. Diese Annahme wird auch als *Eye-Mind-Hypothese* bezeichnet [18]. Ein Problem bei der Analyse von Eye-Tracking-Daten stellt daher das periphere Sehen dar. Teilweise entspricht die gerade fixierte Stelle des Stimulus nicht der wahrgenommenen Information. Dies kann auch der Fall sein, wenn der Betrachter zwar auf einen Punkt fixiert, gleichzeitig aber gedanklich mit etwas anderem beschäftigt ist.

Trotz dieser Schwäche hat sich Eye-Tracking als Indikator für gedankliche Strategien in der Forschung bewährt. Von Studien zur Benutzerfreundlichkeit, etwa von Software und Automobilarmaturen, bis hin zur Marktforschung gibt es weitreichende Anwendungsgebiete [9].

2.2.2 Stimulus

Als Stimulus wird beim Eye-Tracking das Bild bezeichnet, dass vom Probanden während einer Studie gesehen wird. Dabei kann es sich unter anderem um eine Photographie, ein Video, eine Benutzeroberfläche oder die reale Umgebung handeln. In vielen Fällen, etwa bei Usability-Studien, kann der Proband mit der Quelle des Stimulus interagieren und diesen damit verändern. Eine Aufzeichnung des vom Probanden gesehenen Bildes erleichtert die Analyse und Visualisierung der gesammelten Daten.

2.2.3 Techniken

Während die ersten Eye-Tracker noch mit Fotoplatten arbeiteten, wird bei modernen Geräten eine Kamera in Kombination mit infraroter Beleuchtung genutzt, um ein Bild der Augen aufzuzeichnen. Dabei wird das unterschiedliche Reflexionsverhalten der von außen sichtbaren Augenpartien genutzt, um die Pupille und die Hornhaut zu identifizieren. Anschließend kann durch deren Position die Blickrichtung berechnet werden. Vor der Durchführung einer Studie muss das Gerät in der Regel erst für jeden Probanden kalibriert werden. Dabei werden oft nacheinander mehrere Punkte angezeigt, die vom Betrachter zu fixieren sind.

Bei der Auswertung des Kamerabildes und anderer Sensoren, zum Beispiel Pulsmessgeräte oder Elektroenzephalographen, können noch weitere Daten gesammelt werden. Dazu gehört die Erkennung geschlossener Augen, die zur Vermeidung fehlerhaft erkannter Fixationen notwendig ist. Weiterhin kann die Größe der Pupille gemessen werden, welche unter anderem ein Indikator für Erschöpfung oder mentale Belastung ist. Manche Eye-Tracker werten lediglich die Daten für ein Auge aus, während andere binokular messen, um eine höhere Genauigkeit zu erreichen, etwa im dreidimensionalen Raum. Dabei muss beachtet werden, dass die Blickrichtung des einen Auges von der des anderen leicht abweichen kann [3].

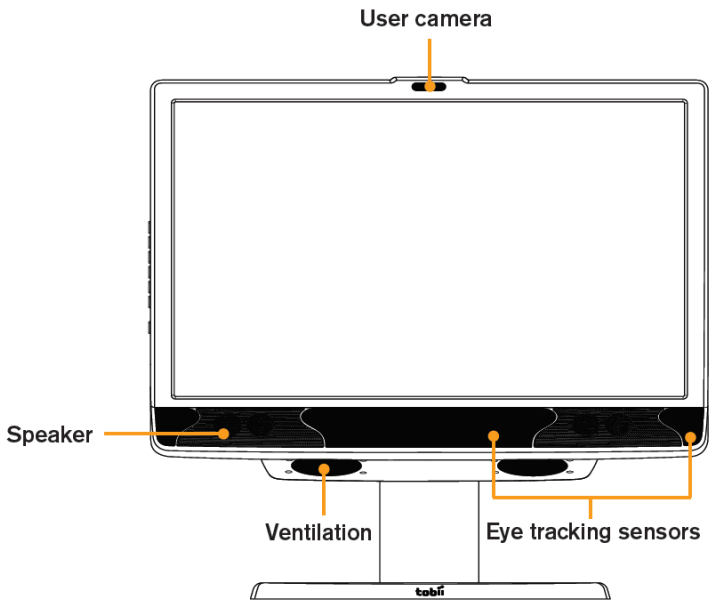


Abbildung 2.3: Schematische Darstellung eines Bildschirm-basierten Eye-Trackers. Dieses Modell besitzt eine integrierte Kamera am oberen Rand zur Aufnahme des Benutzers. Am unteren Rand befinden sich außer den verbauten Lautsprechern die eigentlichen Eye-Tracking Sensoren, also die Kameras, die für die Aufzeichnung der Augenbewegungen zuständig sind. Eine Lüftung schützt die Hardware vor Überhitzung. Die Abbildung entstammt dem Benutzerhandbuch des gezeigten Gerätes und wurde leicht bearbeitet [41].

Usability-Studien für Software werden oft unter Verwendung eines Bildschirm-basierten Eye-Trackers durchgeführt, welcher in Abb. 2.3 veranschaulicht wird. Er zeigt den jeweiligen Stimulus an und enthält andererseits auch die Kamera, sodass eine Abstimmung zwischen den Positionen von Anzeige- und Aufzeichnungsgerät nur einmalig vom Hersteller vorgenommen werden muss. Allerdings sollten Bewegungen des Probanden vermieden werden und der Abstand zwischen Augen und Bildschirm konstant bleiben.

So genannte *Head-Mounted Eye-Tracker* erlauben es dem Probanden, seinen Kopf und sich selbst frei zu bewegen. Dabei wird in der Regel zusätzlich ein Video von der gesehenen Umgebung aufgenommen, um Fixationen später darauf abbilden zu können. Hier kann es auch notwendig sein, die Ausrichtung des Kopfes im Raum zu erfassen, um in Kombination mit den Blickrichtungen der Augen das fixierte Objekt zu berechnen.

Die Qualität der Daten hängt stark von der Leistungsfähigkeit der Hard- und Software ab. Die Abtastfrequenz von Eye-Trackern liegt je nach Modell zwischen 25 und mehreren tausend Hertz. Dem *Nyquist-Shannon Sampling Theorem* folgend muss die Samplingfrequenz mindestens dem Doppelten der maximalen Signalfrequenz entsprechen, um Aliasing-Effekte zu vermeiden. So ist nach einer Aufzeichnung beispielsweise nur bekannt, zwischen welchen Samples eine Sakkade begonnen hat, der Messfehler ist also umgekehrt proportional zur Samplingfrequenz [15]. Nach oder auch schon während der Aufnahme werden die Rohdaten mit den einzelnen Blickpunkten weiterverarbeitet. Mit Hilfe von *Thresholding* wird abgegrenzt, welche davon zu welcher Fixation gehören und wann Sakkaden stattgefunden haben.

2.2.4 Scanpath

Der Begriff *Scanpath* wurde 1971 von David Noton und Lawrence Stark eingeführt, um das Sehverhalten von Probanden bei der Wahrnehmung von Mustern zu beschreiben [31]. Er bezeichnet eine Folge von nach einander ausgeführten Fixationen und den zwischen ihnen liegenden Sakkaden bei der Wahrnehmung von visuellen Stimuli. Es hatte sich herausgestellt, dass ein Betrachter bei der wiederholten Präsentation eines Bildes dessen Details jeweils in einer ähnlichen Reihenfolge fixierte. Dagegen zeigten verschiedene Probanden bei dem selben Bild unterschiedliche Betrachtungsstrategien. Auch zwischen unterschiedlichen Bildern, die von der selben Person gesehen wurden, ergaben sich jeweils andere Scanpaths.

Oft werden Scanpaths durch ihre Fixationen, beziehungsweise deren Positionen, gespeichert und dargestellt. Dabei werden Fixationen durch Kreise oder Symbole repräsentiert, die durch den Sakkaden entsprechende Linien verbunden sind, wie in Abb. 2.4 und 2.5 gezeigt. Eine zusätzliche Visualisierung der Fixationsdauer kann durch eine entsprechende Größe der Kreisflächen umgesetzt werden. Wurde ein Stimulus in Bereiche, wie etwa *Areas Of Interest*, eingeteilt, kann die Position auf den jeweiligen Bereich, in dem die Fixation liegt, verallgemeinert werden (siehe nächsten Abschnitt). Andere Repräsentationsarten sind zum Beispiel Sequenzen von Sakkadenwinkeln und -distanzen oder Fixationsdauern [10].

2.2.5 Area of Interest

Um die Analyse von Eye-Tracking-Daten zu vereinfachen, kann der visuelle Stimulus in Bereiche aufgeteilt werden. Diese werden oft als Area Of Interest, kurz *AOI*, oder *Region Of Interest*, beziehungsweise *ROI*, bezeichnet.

Mit ihrer Hilfe ist es möglich, die Positionen von Vorkommnissen, vor allem von Fixationen, einem Teil des gesehenen Bildes zuzuordnen. Dieser Schritt verringert die Komplexität der gesammelten Daten und vereinfacht damit die weitere Analyse. Durch die Diskretisierung, beziehungsweise *räumliches Downsampling*, ergibt sich allerdings auch das Problem, dass weit auseinander liegende Fixationen innerhalb der selben AOI womöglich gleich behandelt werden. Währenddessen könnten Fixationen, die trotz ähnlicher Position zu verschiedenen AOIs gehören, als unterschiedlich betrachtet werden.

Für die eigentliche Aufteilung eines Stimulus in AOIs gibt es verschiedene Ansätze. Der erste besteht darin, ein Gitter über den Stimulus zu legen, was zu regelmäßigen, jeweils gleich großen Regionen führt. Auch eine manuelle semantische Gliederung des gezeigten Inhalts ist möglich, sofern die Semantik der Bereiche bekannt und abgrenzbar ist. Eine dritte Möglichkeit ist die automatische Erzeugung von AOIs durch Attention-Maps oder Clustering, welche in den Abschnitten 2.3.1 und 2.4.1 behandelt. In den zwei folgenden Abschnitten werden die jeweiligen Vor- und Nachteile sowie mögliche Anwendungsgebiete von gitterförmigen und semantischen AOIs aufgeführt.

Gitterförmige AOIs

Bei dieser Art von AOIs wird der meist rechteckige Stimulus so aufgeteilt, dass sich gleichmäßig verteilte und in ihren Maßen identische Bereiche ergeben, wie Abb. 2.4 zeigt.

Vorteilhaft ist hier vor allem, dass die Erzeugung der AOIs völlig unabhängig vom Stimulus ist, wodurch die Anwendung bei unklarer Semantik oder dynamischen Bilddaten erleichtert wird. Beispiele für Stimuli ohne klar abgrenzbare semantische Bereiche sind Fotografien, Filme oder aufgezeichnete Bewegungen des Probanden in der realen Welt. Damit eignen sich gitterförmige AOIs auch insbesondere für Fälle, in denen eine Nutzung von semantischen AOIs problematisch ist. Da die Festlegung der AOIs automatisch geschieht, wird zudem die Vorbereitung der Analyse vereinfacht, allerdings muss ein sinnvoller Parameter für die *Auflösung*, also die Größe der AOIs, gefunden werden.

Semantische AOIs

Eine semantische Gliederung des Stimulus ist mit einem gewissen Aufwand verbunden. In vielen Fällen müssen die jeweiligen Grenzen manuell festgelegt werden, auch wenn es beispielsweise bei HTML-Dokumenten oder ähnlichen Benutzeroberflächen möglich ist, sie automatisch zu generieren. Abb. 2.5 zeigt eine Webseite mit AOIs über den einzelnen Bestandteilen des Layouts und zwei Scanpaths.

Auch bei Fotografien kann durch eine automatische Objekterkennung auf manuelle Arbeit verzichtet werden, dabei kann jedoch die Qualität der Ergebnisse leiden. Eine weitere Möglichkeit für die Erzeugung der AOIs ist die Benutzung von Scanpaths und Attention Maps mit Hilfe von Grenzwerten

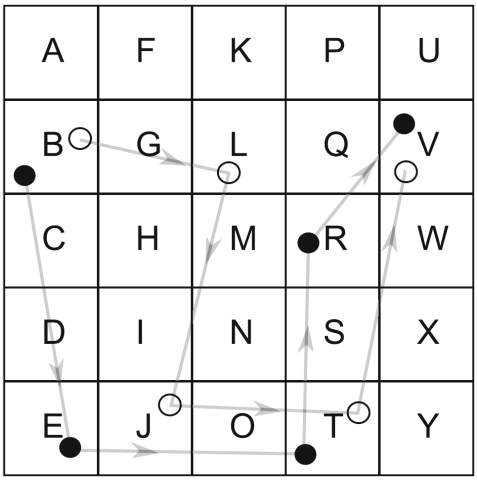


Abbildung 2.4: Ein gitterförmiges Area-of-Interest-Schema mit durch Buchstaben gekennzeichneten AOIs ohne dargestellten Stimulus. Darüber sind zwei Scanpaths zu sehen, die durch Punkte für Fixationen und Linien für die dazwischen liegenden Sakkaden visualisiert sind. Die Pfeile geben dabei die Richtung der Sakkaden und damit die Reihenfolge der Fixationen an. Die beiden Pfade können nach den die Fixationen enthaltenden AOIs durch die Zeichenketten BETRV (schwarz) und BLJTV (weiß) repräsentiert werden [7].

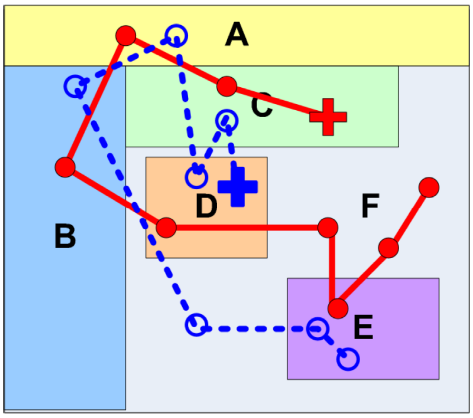


Abbildung 2.5: Zwei Scanpaths als Linien über einem Stimulus visualisiert. Es handelt sich hierbei um eine Webseite, die in semantische AOIs eingeteilt wurde. Beide Scanpaths starten bei den mit einem Plus markierten Fixationen und entsprechen den aus AOI-Aufenthalten zusammengesetzten Zeichenketten CCABDFEFF (durchgezogen, rot) und DCDABFEE (gestrichelt, blau) [10].

oder Clustering, wobei in diesen Fällen die Semantik nur indirekt über das Betrachtungsverhalten der Probanden mit einbezogen wird.

Liegt ein Stimulus vor, bei dem keine klare Abgrenzung zwischen semantisch unterschiedlichen Bereichen existiert, kann sowohl die automatische, als auch die manuelle Festlegung der AOI Grenzen problematisch sein. Dies ist auch der Fall, wenn komplexe oder feine Muster enthalten sind. Auch wenn sich das Bild oft verändert, ist es oft nicht praktikabel, für jeden Zeitpunkt oder Frame die jeweiligen Positionen zu verändern. Insofern eignet sich dieser Ansatz vor allem für klar strukturierte, semantisch eindeutig zuzuordnende und weitgehend statische Inhalte.

Im Gegensatz zu gitterförmigen AOIs ist die Form hier frei wählbar, es sind also auch Ellipsen oder Polygone einsetzbar. Zusätzlich können sich Überschneidungen zwischen verschiedenen Bereichen ergeben, wodurch sich auch *semantische Beziehungen* darstellen lassen können. Zeigt ein Bild beispielsweise mehrere Personen, könnte es sinnvoll sein, für jedes Gesicht eine eigene *Sub-AOI* anzulegen, die innerhalb der AOI der Person liegt und als deren Kind gekennzeichnet ist. Dies könnte man weiter fortsetzen und etwa die Augen als dem Gesicht untergeordnete Bereiche festlegen.

Auch eine Gruppierung von AOIs mit ähnlicher Bedeutung ist möglich. Bei obigem Beispiel wäre eine Gruppe für alle Personen denkbar, sodass bei der späteren Analyse der Daten leicht feststellbar wird, wann der Proband seinen Fokus auf eine Person gelegt hat, unabhängig davon, welche genau es war.

Da ein Stimulus selten komplett von AOIs abgedeckt wird, kann eine *Whitespace-AOI* in Betracht gezogen werden, die alle Fixationen enthält, die keiner anderen zugeordnet werden konnten. Dadurch wird verhindert, dass ein Übergang als direkt fehlinterpretiert wird, obwohl er nicht unmittelbar zwischen zwei inhaltsbezogenen Objekten, sondern über eine oder mehrere Fixationen im Whitespace verlief.

AOI Metriken

Ähnlich zu Fixationen und Sakkaden kommen bei der Verwendung von AOIs Ereignisse vor, die bei der Beschreibung und Analyse des Blickverhaltens auf dem Stimulus nützlich sind [15]. Abb. 2.6 stellt die im Folgenden erläuterten Ereignisse graphisch dar.

AOI-Treffer (Hit). Wenn der Blick des Probanden eine bestimmte AOI fixiert, nachdem direkt vorangegangene Fixationen außerhalb von ihr stattfanden, entspricht dies einem AOI-Treffer. Ein erneuter Treffer kann dementsprechend nur auftreten, wenn die AOI verlassen und sie oder eine andere erneut fixiert wird. Durch geeignete Filter kann festgelegt werden, dass ein Treffer erst nach einer gewissen Aufenthaltsdauer gültig wird.

AOI-Aufenthalt (Dwell oder Gaze). Ein Aufenthalt dauert an, solange alle weiteren Fixationen nach einem Treffer in der selben AOI landen und endet mit dem Verlassen der AOI. Alle Fixationen dazwischen werden diesem Aufenthalt zugeordnet, seine Dauer ist die Summe aller enthaltenen Fixations- und Sakkadendauern und wird als *Dwell-Time* bezeichnet.

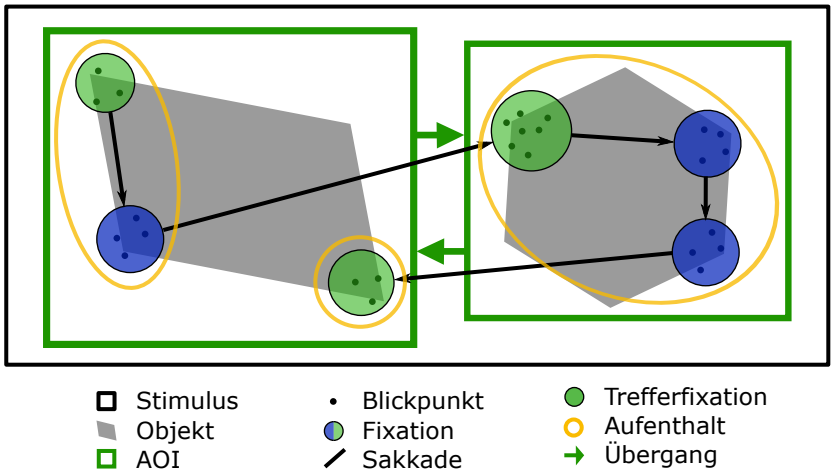


Abbildung 2.6: Diese Zeichnung zeigt einen Stimulus mit zwei Objekten, die jeweils in einer rechteckigen AOI liegen. Nachdem Blickpunkte zu Fixationen und dazwischen liegenden Sakkaden zusammengefasst wurden, können die Ereignisse *Treffer*, *Aufenthalt* und *Übergang* festgestellt werden. Treffer werden dabei durch die erste Fixation in einer neuen AOI definiert, Aufenthalte zusätzlich durch alle weiteren folgenden Fixationen in der selben AOI. Übergänge finden statt, wenn die nächste Fixation in einer anderen AOI liegt.

AOI-Übergang (Transition). Übergänge finden statt, wenn eine neue Fixation in einer anderen AOI liegt als die vorherige. So definieren die letzte Fixation der verlassenen AOI und die gerade stattgefundene Fixation in der neuen AOI gemeinsam einen Übergang zwischen diesen beiden AOIs.

2.3 Vergleich von Scanpaths

Für die Ähnlichkeit mehrerer Scanpaths gibt es verschiedene Kriterien. Generell sollte die allgemeine Form und Position der Pfade und die zeitliche Reihenfolge der einzelnen Teilpfade übereinstimmen. Auch die Dauer der Fixationen sollte nicht zu stark abweichen [7]. Dabei können auch Pfade als ähnlich betrachtet werden, die bei ähnlicher Form eine unterschiedliche Skalierung haben. Diese kann sowohl räumliche, als auch zeitliche Ausmaße betreffen. Hat ein Pfad eine ähnliche Form, aber den umgekehrten zeitlichen Verlauf, kann er trotzdem als gleichartig angesehen werden. Das ist dann der Fall, wenn zwei Probanden den selben Pfad betrachtet haben, jedoch in entgegengesetzter Richtung.

Alle Kriterien können auch auf Teilpfade angewandt werden, sodass zwei Scanpaths anhand der Vorkommen gemeinsamer *Muster* verglichen werden. Es kann beispielsweise vorkommen, dass bei der Lösung einer Aufgabe eine bestimmte Teilaufgabe mit derselben Strategie gelöst wurde, obwohl sich der Rest der Lösung unterscheidet. Hier besteht also eine *partielle Übereinstimmung*.

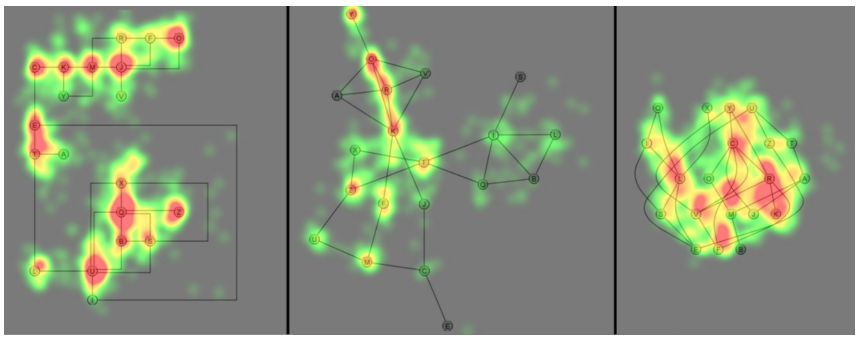


Abbildung 2.7: Beispiel dreier Attention-Maps, die als Heat-Maps visualisiert wurden. In dieser Studie wurde die Lesbarkeit von Graph-Layouts untersucht. Rot steht für Stellen mit hoher Aufmerksamkeit, Gelb und Grün für geringere und graue Stellen wurden überhaupt nicht betrachtet [34].

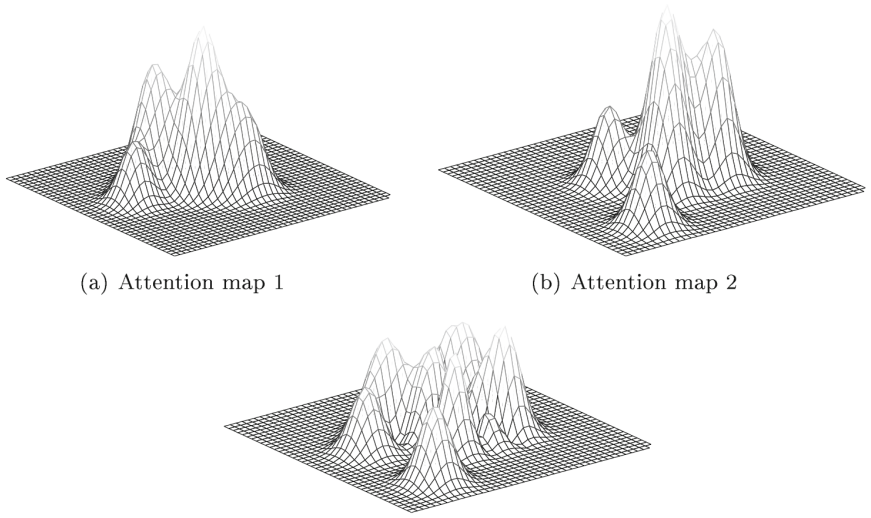
In diesem Abschnitt werden verschiedene Ansätze zum Vergleich von Scanpaths vorgestellt und verglichen. Attention-Maps, Markov-Modelle und Fixations-basierte Ansätze konzentrieren sich auf räumliche Daten und beziehen zeitliche Informationen nicht oder nur teilweise ein. Vektor- und String-basierte Ansätze beziehen räumliche und zeitliche Daten ein, auch wenn diese teilweise quantisiert und dadurch ungenauer werden. In diesem Kapitel werden Vektor-basierte Ansätze nicht behandelt, ein Beispiel dafür wurde in MultiMatch [7, 17] umgesetzt, auf das im Abschnitt 4.2 ausführlich eingegangen wird.

2.3.1 Attention- und Dwell-Maps

Attention-Maps, auch *Saliency-Maps* genannt, können als eine Funktion über dem Stimulus betrachtet werden, die für jeden Punkt angibt, wie lange er vom Betrachter fixiert wurde. Zur Visualisierung bietet sich eine *Heat-Map* an, ein Beispiel dafür zeigt Abb. 2.7.

Eine *Dwell-Map* entspricht einer durch Areas of Interest quantisierten Attention-Map. Sie besteht aus einer Auflistung aller AOIs mit ihren zugehörigen summierten Aufenthaltszeiten, also der Zeit, in der ein Proband diese AOI insgesamt angesehen hat. Mögliche Darstellungen sind etwa eine zweispaltige Tabelle mit den Bezeichnungen der AOIs in einer Spalte und den Zeiten in der anderen oder eine Heat-Map je einer einfarbig ausgefüllten Fläche pro Area of Interest.

Durch Bildung der mathematischen Differenz zweier Maps kann die Ähnlichkeit festgestellt werden, wie in Abb. 2.8 gezeigt [50]. Außerdem gibt es mit der *Kullback-Leibler-Divergenz*, der *Receiver-Operating-Characterisitc-Analysis* und der Earth-Mover-Distanz [6] noch weitere Methoden zur Berechnung der Ähnlichkeit von Saliency-Maps [22]. Diese Vergleiche betrachten jedoch alle nur die räumliche Verteilung der Aufmerksamkeit. Zeitbezogene Informationen werden nur in ihrer Summe betrachtet und sequentielle Daten ignoriert. Es handelt sich also um eine sogenannte *No-History-Analysis* [15].



(a) Attention map 1 (b) Attention map 2

(c) Attention map difference

Abbildung 2.8: Zwei Attention-Maps (a, b) und ihre Attention-Map-Differenz (c) [7].

2.3.2 Markov-Modelle

Ein probabilistischer Ansatz zum Vergleich mehrerer Probanden sind *Markov Modelle*. Sie beschreiben die Übergangswahrscheinlichkeiten zwischen AOIs und können mehrere Stufen besitzen [15]. Bei einem Markov-Modell nullter Ordnung entsprechen die Werte denen einer Dwell-Map und stehen für die Wahrscheinlichkeit, dass die jeweilige AOI als nächste besucht wird. Wird auch die zuvor besuchte AOI in Betracht gezogen, entsteht ein Modell erster Ordnung. Mit jeder weiteren Stufe kommt ein weiterer Übergang hinzu, allerdings werden Modelle mit einer Ordnung höher als zwei äußerst selten zur Analyse von Eye-Tracking-Daten benutzt.

Da hier auch die zeitliche Abfolge eine Rolle spielt, wird dieser Ansatz als *Short-History-Analysis* bezeichnet [15].

Eine *Full-History-Analysis* hingegen muss den vollständigen gemessenen Zeitraum mit allen sequenziellen Informationen miteinbeziehen, vergleicht also zwei oder mehr Scanpaths in ihrer Gesamtheit von räumlichen und sequenziellen Daten.

2.3.3 Fixations-basierte Ansätze am Beispiel der Mannan-Distanz

Die Mannan-Distanz wurde direkt für den Vergleich von Scanpaths entwickelt [26, 27, 28]. Bei dieser Metrik werden nur räumliche Informationen mit einbezogen, die zeitliche Reihenfolge der Fixationen und deren Dauer werden also nicht betrachtet [22]. Die Ähnlichkeit wird untersucht, indem für alle Fixationen aus einem Scanpath der Abstand zwischen ihr und der jeweils nächstliegenden im anderen Scanpath berechnet wird. Dieser Vorgang wird in beide Richtungen, also von beiden Scanpaths ausgehend durchgeführt.

Zur Gewichtung der Ähnlichkeit werden zufällig erzeugte Scanpaths gleicher Länge genutzt und ein Ähnlichkeitsindex I_s berechnet [22]:

$$I_s = \left[1 - \frac{D}{D_r}\right] \times 100 \tag{2.1}$$

Dabei ist D_r der Abstand zwischen zwei zufällig generierten Fixationsmengen und D die Distanz zwischen den zu vergleichenden Scanpaths. Diese Abstände werden folgendermaßen errechnet:

$$D^2 = \frac{n_1 \sum_{j=1}^{n_2} d_{2j}^2}{2n_1n_2(a^2+b^2)} + \frac{n_2 \sum_{i=1}^{n_1} d_{1i}^2}{2n_1n_2(a^2+b^2)} \tag{2.2}$$

Hierbei sind n_1, n_2 die Anzahl der Fixationen beider Scanpaths, d_{1i}, d_{2j} stehen für den Abstand zwischen der i-ten beziehungsweise j-ten Fixation in einen Scanpath und der am nächsten gelegenen im anderen, die Indizes 1 und 2 geben an, aus welchem Scanpath die betrachtete Fixation stammt. Die Variablen a, b entsprechen den räumlichen Maßen des Stimulus. In Worten gefasst berechnet Formel 2.2 die Summe der Abstände zwischen den am nächsten liegenden Paaren von Fixationen und normalisiert diese zur Anzahl der Fixationen und Größe des Stimulus. Dieser Wert wird von jedem der Pfade ausgehend berechnet und die beiden Werte addiert. Ergebnisse des Algorithmus reichen von 0 für *zufällige Übereinstimmung* bis 100 für *identische Scanpaths*.

Ein Vorteil dieses Ansatzes ist die direkte Nutzung von Fixationspositionen anstatt von durch AOIs stärker quantisierten Daten. Zudem entfällt die Notwendigkeit und Problematik der Festlegung von AOIs und ihrer Grenzen. Der große Nachteil ist die erwähnte Nichtbetrachtung zeitlicher Daten, wodurch Scanpaths mit ähnlicher Form, aber unterschiedlicher Reihenfolge der Fixationen als ähnlich betrachtet werden. Es handelt sich also wie bei Attention-Maps um eine No-History-Analysis. Das kann in vielen Szenarien ausreichend sein, jedoch unter Umständen dann nicht, wenn etwa die Strategie eines Probanden bei der Lösung einer Aufgabe untersucht werden soll. Ein Scanpath mit zeitlich genau umgekehrten Fixationen würde hier als identisch gewertet, auch wenn die Strategie sich unterscheidet. Im Extremfall könnten alle Fixationen des einen Scanpaths einer einzigen des anderen Scanpaths zugeordnet werden, was zu einem völlig falschen Ergebnis führen würde [7]. Ein weiteres Problem bei diesem Ansatz ist, wie bei anderen auch, dass die Qualität des Ergebnisses unter einer großen Varianz in der Länge mehrerer Scanpaths leidet. Aufgrund dieser Mängel und der Verfügbarkeit verbesserter Varianten, etwa mit einem erzwungenen Eins-zu-Eins-Matchings der Fixationen, wird die Mannan-Distanz kaum mehr zum Vergleich von Scanpaths eingesetzt.

2.3.4 String-basierte Ansätze

Im Folgenden werden einige gebräuchliche Ansätze zum Vergleich von Strings vorgestellt. Eine umfassendere Auflistung ist beispielsweise bei Gomaa und Fahmy [12] zu finden. Diese Gruppe von Lösungsansätzen arbeitet auf Basis von Zeichenketten, die im Falle von Scanpaths meist Serien von Fixationen repräsentieren. Obwohl die meisten Algorithmen auch die Gemeinsamkeiten, also ein Matching zwischen den Strings berechnen, ist für die Berechnung eines Ähnlichkeitswertes von Scanpaths im Grunde nur die Größe dieses Matchings von Bedeutung.

Hamming Distanz

Die Hamming-Distanz [13] wird unter anderem für die Fehlerkorrektur bei Übertragungen benutzt. Sie ist eine eher beschränkte Vergleichsmetrik, da bei ihrer Anwendung beide Strings die gleiche Länge haben müssen. Der Wert für die Distanz ergibt sich aus der Anzahl der unterschiedlichen Stellen. So beträgt die Hamming-Distanz zwischen *100111* und *10**11**0**1* zwei, da sich die Codewörter in zwei Bits, hier fett markiert, unterscheiden.

Levenshtein-Distanz

Oft auch als *String-Edit-Distanz* bezeichnet, berechnet diese Metrik den Abstand zwischen zwei Zeichenketten, indem Kosten für das Einfügen, Löschen und Ersetzen eines Symbols festgelegt werden [23]. Dann werden die minimalen Kosten gesucht, die nötig sind, um mit diesen drei Operationen den einen String in den anderen umzuformen.

In der Regel werden für Einfügen und Löschen jeweils Kosten in Höhe von 2 veranschlagt, während eine Ersetzung 1 kostet. Es sind allerdings auch Funktionen denkbar, welche abhängig von der Art oder Position des Zeichens unterschiedliche Kosten berechnen [33]. Ein Beispiel dafür ist die *Schreibmaschinendistanz*, hier sind die Kosten abhängig von der Entfernung der Tasten.

Um eine Berechnung effizient und ohne Redundanz durchzuführen, wird dynamische Programmierung genutzt. Für die Ermittlung der Distanz zweier Strings A und B dient eine Tabelle der Größe $(n+1) \times (m+1)$ als Zwischenspeicher, wobei $n = |A|$ und $m = |B|$ die jeweiligen Längen der Strings sind. Tabelle 2.2 zeigt eine solche Tabelle nach der beispielhaften Berechnung der Levenshtein-Distanz zwischen *abbcba* und *aabbc*. In jedem Feld stehen nach der Ausführung des Algorithmus die minimalen Kosten zur Umwandlung eines Teilstrings in den anderen, wobei diese Strings gebildet werden, indem die Länge jeweils dem Zeilen- beziehungsweise Spaltenindex abzüglich 1 entspricht.

Zur Vorbereitung wird die erste Zeile mit den Einfüge- und die erste Spalte mit den Löschkosten initialisiert. Danach wird jede verbleibende Zelle mit folgender Funktion basierend auf den bereits berechneten Werten ausgefüllt:

$$T[i,j] = \min \begin{cases} T[i-1,j] + Kosten_{\text{Einfügen}} & \\ T[i-1,j-1], & \text{falls} A_i = B_j \\ T[i-1,y-1] + Kosten_{\text{Ersetzen}}, & \text{sonst} \\ T[i,j-1] + Kosten_{\text{Löschen}} & \end{cases} \tag{2.3}$$

		a	b	b	c	b	a
	0	2	4	6	8	10	12
a	2	0	2	4	6	8	10
a	4	2	1	3	5	7	9
b	6	4	3	1	2	4	6
b	8	6	5	3	2	4	6
c	10	8	7	5	3	3	**5**

Tabelle 2.2: Beispieltabelle für die Berechnung der Levenshtein-Distanz zwischen den beiden Worten *abbcba* und *aabbc* mit den Kosten 1 für Ersetzen und 2 jeweils für Einfügen und Löschen. Das Ergebnis steht in der Zelle unten rechts, es kommt durch drei Ersetzungen und eine Einfügung zustande (alle grau hinterlegt, von links oben nach recht unten). Diese können durch eine Rückverfolgung gefunden werden, wenn ausgehend vom Ergebnis nachgesehen wird, woher jeweils das Minimum stammte.

Durch Normalisieren des Ergebnisses d mit der maximalen Scanpathlänge, gefolgt von Spiegelung an der Null und Verschiebung um Eins kann ein Resultat im Bereich zwischen Null und Eins gewonnen werden, bei dem Null für komplett verschiedene und Eins für identische Strings steht [15]:

$$\widehat{d} := 1 - \frac{d}{max(m,n)} \tag{2.4}$$

Damerau-Levenshtein-Distanz

Die Damerau-Levenshtein-Distanz [5, 43] erweitert den Standard-Levenshtein-Algorithmus um eine Behandlung von in der Position vertauschter Zeichen, etwa *AB* und *BA*. Während bei Levenshtein in diesem Fall zwei Operationen nötig wären, wird hier eine Vertauschung angewandt.

Beim Algorithmus ändert sich dabei lediglich die Berechnung der Kosten in der Tabelle, bei der im Falle einer kreuzweisen Übereinstimmung an den Positionen *i* und *j* nun auch das Feld $T[i-2, j-2]$ berücksichtigt wird:

$$T[i,j] = \min \begin{cases} \min \begin{cases} Levenshteinkosten & \text{falls } i,j > 1 \text{ und } A_i = B_{j-1} \\ T[i-2,j-2] + c_{\text{ Vertauschen}} & \text{und } A_{i-1} = B_j \end{cases} \\ Levenshteinkosten & \text{sonst} \end{cases} \tag{2.5}$$

Needleman-Wunsch-Algorithmus

Um die Ähnlichkeit von Aminosäure-Sequenzen verschiedener Proteine bestimmen zu können, entwickelten Saul B. Needleman und Christian D. Wunsch 1969 einen Algorithmus, der ein globales Matching zweier Zeichenketten berechnet [30].

Dieser auf dynamischer Programmierung basierende Algorithmus sucht einen Pfad mit optimalen Kosten mittels eine Tabelle mit allen möglichen Matchings. Dabei wird versucht, einen maximalen

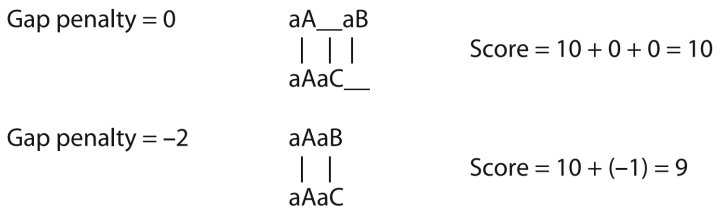


Abbildung 2.9: Auswirkungen der Gap Penalty auf das Ergebnis des Algorithmus. Punkte für Ersetzungen entsprechen denen aus Tabelle 2.3 . Bei einer Gap Penalty von -2 wird auf ein Matching von Lücken zugunsten eines Zeichenmatchings mit -1 Punkten verzichtet [4].

	aA	aB	aC
aA	10	-1	-5
aB	-1	10	-1
aC	-5	-1	10

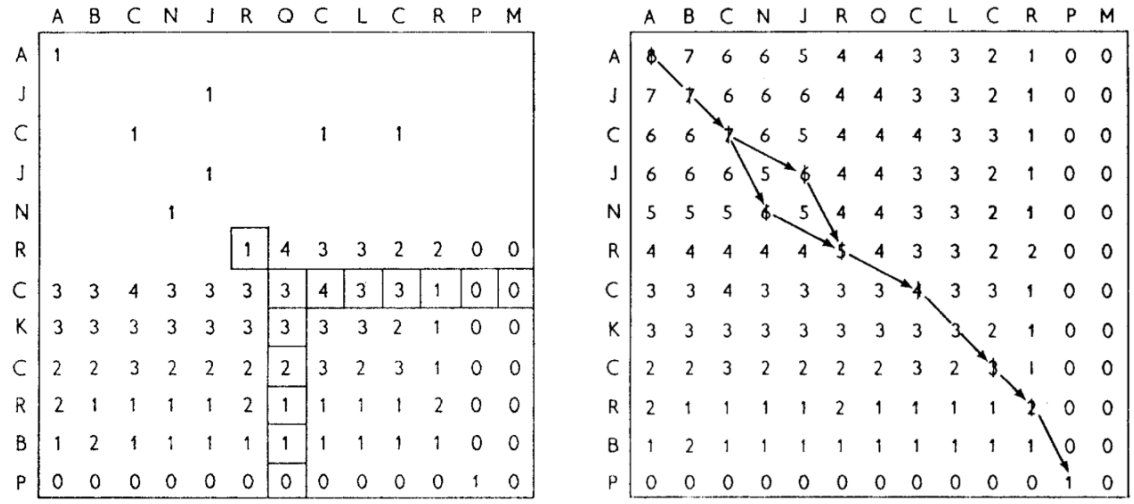
Tabelle 2.3: Beispiel für die Ersetzungsmatrix beim Algorithmus von Needleman und Wunsch. Sie enthält eine Punktzahl für die Ersetzung eines Zeichens durch ein anderes. Die negativen Werte bestrafen eine Ersetzung, etwa $aA \rightarrow aC$. In diesem Beispiel ist die Matrix symmetrisch, sodass die Richtung der Ersetzung keine Rolle bei der Berechnung der Ähnlichkeit spielt [4].

Wert zu erreichen, indem die Zuordnung zu Zeichenpaaren mit Punkten bewertet wird. Für Lücken im Pfad wird eine *Gap-Penalty* genannte Strafe angerechnet, beispielsweise -1. Je kleiner diese ist, desto eher wird auf Lücken verzichtet und stattdessen eine schlechte Zuordnung vorgezogen. Die Auswirkungen der Gap Penalty werden durch Abb. 2.9 verdeutlicht.

Zum Vergleich zweier Zeichen, im ursprünglichen Ansatz Stellvertreter für Aminosäuren, wird eine Ersetzungsmatrix verwendet, ein Beispiel ist in Tabelle 2.3 zu sehen. Die Matrix enthält für jedes mögliche Zeichenpaar einen Wert für dessen Ähnlichkeit, der angibt, wie viele Punkte für eine hypothetische Ersetzung vergeben werden. Eine höhere Zahl in einer Zelle steht für eine größere Ähnlichkeit zwischen den beiden Zeichen, es werden also mehr Punkte vergeben, wenn diese einander zugeordnet werden.

Der Algorithmus berechnet das maximale Matching, indem die Tabelle von unten rechts nach oben links Zeile für Zeile abgearbeitet wird, siehe Abb. 2.10. Für jede einzelne Zelle $[i, j]$ wird der Wert aus den bereits vorhandenen Werten der Zeile $[i + 1]$ und der Spalte $[j + 1]$ berechnet. Aus ihnen wird der maximale Wert genommen und zur Punktzahl der aktuell betrachteten Zelle addiert. Diese Punktzahl wird in der Ersetzungsmatrix nachgeschaut, hängt also von den beiden zu dieser Zelle gehörenden Zeichen ab.

Nach dem vollständigen Ausfüllen der Tabelle wird der optimale Pfad gesucht. Das ist derjenige mit der maximalen Punktzahl, je höher diese ist, desto ähnlicher sind die beiden verglichenen Strings. Er wird gefunden, indem zurückverfolgt wird, woher der maximale Wert des Berechnungsschritts kam, angefangen beim Maximum der Tabelle. Für jeden Abschnitt des Pfades beginnend bei $[i, j]$



	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	1												
J					1								
C			1					1		1			
J					1								
N				1									
R						1	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

	A	B	C	N	J	R	Q	C	L	C	R	P	M
A	8	7	6	6	5	4	4	3	3	2	1	0	0
J	7	7	6	6	6	4	4	3	3	2	1	0	0
C	6	6	7	6	5	4	4	4	3	3	1	0	0
J	6	6	6	5	6	4	4	3	3	2	1	0	0
N	5	5	5	6	5	4	4	3	3	2	1	0	0
R	4	4	4	4	4	5	4	3	3	2	2	0	0
C	3	3	4	3	3	3	3	4	3	3	1	0	0
K	3	3	3	3	3	3	3	3	3	2	1	0	0
C	2	2	3	2	2	2	2	3	2	3	1	0	0
R	2	1	1	1	1	2	1	1	1	1	2	0	0
B	1	2	1	1	1	1	1	1	1	1	1	0	0
P	0	0	0	0	0	0	0	0	0	0	0	1	0

Abbildung 2.10: Links: Ein Beispiel für die Berechnung der Ähnlichkeit bei Needleman und Wunsch. Hier wurde 1 Punkt vergeben, falls zwei Zeichen identisch sind, sonst 0 Punkte. Der Algorithmus geht zeilenweise von unten rechts nach oben links durch die Tabelle. Dabei wird für die aktuell betrachtete Zelle (das oberste eingerahmte Kästchen) seine eigene Punktzahl, also aufgrund der Übereinstimmung $R = R$ in diesem Fall 1, zur maximalen Punktzahl der anderen umrandeten Zellen addiert. Diese wird vor der Betrachtung je nach Entfernung noch um eine Gap-Penalty verringert. Rechts: Die ausgefüllte Tabelle. In diesem Fall gibt es zwei mögliche Pfade für das maximale Matching. Dieses endet im größten Wert der ersten Zeile oder Spalte [30].

werden wieder die Zeile $[i + 1]$ und die Spalte $[j + 1]$ nach ihrem Maximum durchsucht. Dabei wird zur Berücksichtigung der eventuell entstehenden Lücke beim Matching eines der Strings eine dem Abstand des verwendeten Wertes entsprechende Gap-Penalty berechnet. In der Zelle des gewählten Wertes beginnt dann der nächste Abschnitt, bis der Rand der Tabelle erreicht wurde. Es können, wie im verwendeten Beispiel, mehrere maximale Matchings existieren, der entsprechende maximale Wert in der Berechnungstabelle, der das Ende des Pfades markiert, ist bei diesen jedoch derselbe.

Smith-Waterman-Algorithmus

Der Smith-Waterman-Algorithmus [40] ist eine Variation des Algorithmus von Needleman und Wunsch, er sucht eine lokale Übereinstimmung der Strings. Inzwischen gibt es Varianten des Smith-Waterman-Algorithmus mit verbesserter Genauigkeit [1] oder Alternativen mit höherer Geschwindigkeit wie *BLAST* [2]. Auch dieser Ansatz arbeitet auf Basis dynamischer Programmierung. Im Gegensatz zu Needleman-Wunsch werden die erste Zeile und die erste Spalte der Berechnungstabelle

mit Nullen initialisiert. Danach wird die restliche Tabelle nach folgenden Regeln gefüllt (Formel übernommen aus [49] und angepasst):

$$T(i,j) = \max \begin{Bmatrix} 0 & \text{(a)} \\ T(i-1,j-1) + E(A_i, B_j) & \text{Match/Mismatch (b)} \\ \max_{k \geq 1}\{T(i-k,j) + W_k\} & \text{Löschung (c)} \\ \max_{l \geq 1}\{T(i,j-l) + W_l\} & \text{Einfügung (d)} \end{Bmatrix} \tag{2.6}$$

Dass zusätzlich über Null maximiert wird, ist eine weitere Modifikation zum Algorithmus von Needleman und Wunsch. $E(a,b)$ hingegen entspricht der dort verwendeten Ersetzungsmatrix. Der zurückgegebene Wert ist umso höher, je ähnlicher a und b sind. W_i dient zur negativen Wertung von Lücken und kann beispielsweise als $-i$ gewählt werden. Dann steigt der Betrag der negativen Wertung, etwa beim Fall *Löschung* mit der Entfernung k des Tabellenfelds $T[i-k,j]$, aus dem der Wert bezogen wird. Zusammengefasst berechnet der Algorithmus für jede Zelle das Maximum aus

(a) Null

(b) der Summe aus der diagonal links oben liegenden Zelle und dem Ähnlichkeitswert

(c) der Löschung von unterschiedlich vielen Zeichen

(d) der Einfügung von unterschiedlich vielen Zeichen.

Im letzten Feld der Tabelle steht nach Durchführung des Algorithmus der größte Wert einer Übereinstimmung zwischen zwei Teilstrings von *A* und *B*.

Longest Common Subsequence

Gesucht wird hier die längste Sequenz von Zeichen, die in beiden Strings enthalten ist [38]. Dieses Problem wird als Longest Common Subsequence, kurz LCS, bezeichnet. Auch dieser Ansatz kommt aus der Bioinformatik und dient dem Auffinden von Ähnlichkeiten zwischen Proteinen oder Ribonukleinsäuren. Es sollen bestimmte Löschungs- und Einfügungsbeschränkungen erfüllt werden, die für die Genetik wichtige Bedingungen widerspiegeln. Inzwischen wird er aber auch unter anderem in der Versionsverwaltung verwendet. Im Gegensatz zum Longest Common *Substring* müssen die Zeichen der LCS in den beiden Eingabestrings nicht direkt hintereinander auftreten. Als einfaches Beispiel sei als Eingabe *ABDCA* und *ACDA* gegeben. Dann sind die LCS dieser zwei Zeichenketten *ACA* und *ADA*. Folgende Vorschrift ermöglicht das Finden der LCS zweier Strings mittels dynamischer Programmierung (Formel übernommen aus [48] und angepasst).

$$LCS\left(A_i, B_j\right) = \begin{cases} \emptyset & \text{falls } i = 0 \text{ oder } j = 0 \\ LCS\left(A_{i-1}, B_{j-1}\right) \frown a_i & \text{falls } a_i = b_j \\ \text{längste}\left(LCS\left(A_i, B_{j-1}\right), LCS\left(A_{i-1}, B_j\right)\right) & \text{falls } a_i \neq b_j \end{cases} \tag{2.7}$$

Für die erste Zeile und Spalte entspricht die LCS der leeren Menge $\emptyset$. Jedes weitere Feld der Tabelle wird dann abhängig von den zwei dazu gehörenden Zeichen der Strings berechnet. Sind diese identisch, wird die LCS entsprechend erweitert. $LCS\left(A_{i-1}, B_{j-1}\right) \frown a_i$ steht hierbei für eine Konkatenation

der bisherigen LCS im Feld links oben vom aktuellen mit dem i-ten Zeichen des Strings A. Bei ungleichen Zeichen wird die bisher längste Teilsequenz beibehalten, welche sich im Feld links vom oder über dem aktuellen befindet. Für die bloße Berechnung der Ähnlichkeit von Scanpaths würde es ausreichen, die Länge der LCS zu kennen. Diese kann mit dem im Listing 2.1 gezeigten Algorithmus berechnet werden.

```
1   function LCSLength(X[1..m], Y[1..n])
2     C = array(0..m, 0..n)
3     for i := 0..m
4       C[i,0] = 0
5     for j := 0..n
6       C[0,j] = 0
7     for i := 1..m
8        for j := 1..n
9           if X[i] = Y[j]
10             C[i,j] := C[i-1,j-1] + 1
11          else
12             C[i,j] := max(C[i,j-1], C[i-1,j])
13    return C[m,n]
```

Listing 2.1: Algorithmus zur Berechnung der Länge der Longest Common Subsequence zweier Strings. Es werden zunächst die erste Zeile und Spalte der Tabelle mit Nullen initialisiert. Danach wird diese zeilenweise von oben links nach unten rechts durchlaufen und für jedes Feld die Länge der aktuellen Longest Common Subsequence berechnet. Analog zur Berechnung der LCS selbst wird bei zwei gleichen Zeichen die Länge inkrementiert, während bei Ungleichheit die bis zu diesem Zeitpunkt größte Länge beibehalten wird [48].

2.4 Visualisierung von Ähnlichkeiten zwischen Daten

Visualisierungen helfen bei der Analyse komplexer Daten, indem sie diese übersichtlich und strukturiert präsentieren. Im Folgenden wird *Clustering* als Technik zur Gruppierung von Daten vorgestellt. Anschließend werden mit *Dendrogrammen* und *Tree-Maps* zwei für hierarchische Cluster geeignete Visualisierungen gezeigt.

2.4.1 Clustering

Zum besseren Verständnis von Ergebnissen ist es oft hilfreich, ähnliche Daten zu gruppieren. Clustering-Algorithmen erledigen diese Aufgabe automatisch, sie können nach Art der Gruppierung in *flache* und *hierarchische* Ansätze eingeteilt werden.

Ein Beispiel für flaches Clustering ist der *k-means-Algorithmus* [24]. Bei ihm werden, je nach gewünschter Gruppenzahl, aus den einzelnen Daten zufällige Centroide gewählt. Alle übrigen Daten werden dann einer Gruppe mit dem am nächsten liegenden, beziehungsweise ähnlichsten, Centroid zugewiesen. Danach wird für jede Gruppe das Element als neues Centroid genommen, für das

die Summe der Abstände zu allen anderen Gruppenmitgliedern minimal ist. Anschließend werden die Gruppen aufgelöst und die letzten beiden Schritte mit den jeweils neuen Centroiden solange wiederholt, bis ein Abbruchkriterium erreicht wurde.

Hierarchisches Clustering erzeugt zunächst eine Hierarchie der Daten, die dann auf einem beliebigen Niveau abgeschnitten werden kann, um Gruppen zu erzeugen. Dabei gibt es sowohl *Top-Down-*, als auch *Bottom-Up-Ansätze*. Während bei ersteren zu Beginn nur eine Gruppe existiert, die dann nach und nach aufgeteilt wird, werden bei letzteren schrittweise Gruppen verschmolzen. An dieser Stelle wird nur auf eine häufig verwendete, spezielle Form des Bottom-Up-Clustering eingegangen, die als *hierarchisches agglomeratives Clustering*, abgekürzt HAC, bezeichnet wird [29]. Dieses funktioniert folgendermaßen:

- Zunächst wird für jedes einzelne Element eine eigene Gruppe erzeugt.
- Dann werden immer die beiden jeweils ähnlichsten Gruppen zu einer gemeinsamen neuen Gruppe hinzugefügt.
- Das wird so lange wiederholt, bis eine festgelegte Anzahl an Gruppen erreicht ist, die verbleibenden Gruppen zu unterschiedlich sind oder nur noch eine Gruppe übrig geblieben ist.

Der Grad der Ähnlichkeit kann dabei verschiedenartig berechnet werden. Eine Möglichkeit dafür ist der Durchschnitt der Abstände aller möglichen Paare von Elementen, bei denen je eines aus beiden Clustern genommen wird. Außerdem können die zwei zueinander ähnlichsten oder unterschiedlichsten Elemente aus zwei Gruppen als deren Stellvertreter dienen. Auch die Wahl eines Gruppen-Centroids als stellvertretendes *Most-Central-Element* ist möglich. Hierfür könnte etwa jenes dienen, das allen anderen Gruppenmitgliedern am ähnlichsten ist. Um redundante Berechnungen zu sparen, kann eine Matrix mit den jeweiligen Abständen zwischen allen zu clusternden Daten genutzt werden.

2.4.2 Dendrogramm

Auch bei Clustern kann eine geeignete Visualisierung die Analyse der Daten erheblich vereinfachen. Für eine Hierarchie, also einen *Baum* mit Wurzel, bietet sich in vielen Fällen ein Dendrogramm an, wie es in Abb. 2.11 zu sehen ist.

Es besteht aus Beschriftungen für die vorhandenen Elemente und Linien, die für die Darstellung der hierarchischen Beziehungen zuständig sind. Eine Verbindung zwischen zwei Elementen oder Gruppen steht für die relative Ähnlichkeit, beispielsweise für die Vereinigung dieser beiden beim Clustering. So zeigt ein Dendrogramm eines geclusterten Datensatzes von unten nach oben die Reihenfolge der Gruppenvereinigungen an, womit gleichzeitig auch gezeigt wird, welche beiden Gruppen jeweils in jedem Schritt des Clustering-Algorithmus am ähnlichsten waren. Bei einer großen Anzahl von verarbeiteten Elementen kann die Hierarchie aufgrund ihrer Größe, vor allem in der Breite, unübersichtlich werden. Kreisförmige Layouts versuchen dem entgegenzuwirken und trotzdem die Lesbarkeit zu erhalten.

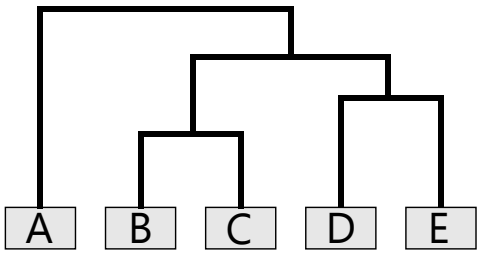


Abbildung 2.11: Ein Cluster dargestellt als Dendrogramm. Unten befinden sich die mit Buchstaben beschrifteten Elemente, die durch das Clustering hierarchisch gruppiert wurden. Die Reihenfolge der Aktionen kann von unten nach oben abgelesen werden. Je ähnlicher zwei Gruppen sind, desto früher wurden sie verschmolzen, was durch eine sie verbindende Linie gezeigt wird. Die erzeugten Gruppen sind $\{B, C\}$, $\{D, E\}$, $\{B, C, D, E\}$ und $\{A, B, C, D, E\}$. In diesem Beispiel wurde erst gestoppt, als nur noch eine Gruppe übrig war.

2.4.3 Erweiterbare Baumansicht

Eine interaktive Darstellung einer Hierarchie stellen Baumansichten dar, die beispielsweise in Dateimanagern genutzt werden. Zunächst werden nur die obersten Elemente der Hierarchie angezeigt, wie etwa Wurzelverzeichnisse, was einen anfänglichen Überblick vermittelt. Je nach Interesse kann der Benutzer dann eines oder mehrere der Verzeichnisse meist durch Klicken auf eine Schaltfläche mit einem Pluszeichen erweitern, wodurch auch deren untergeordnete Elemente gezeigt werden. Umgekehrt kann ein Verzeichnis durch einen Klick auf ein Minuszeichen reduziert werden. Um zu verdeutlichen, welche Elemente einem anderen untergeordnet sind, können Linien oder Einrückung verwendet werden.

Nicht benötige Daten werden in dieser Visualisierung zu einem großen Teil verborgen, was die kognitive Last verringert und die Suche nach einem Zielelement beschleunigt. Andererseits ist ein gewisser Aufwand nötig, um ein Element zu finden, dass sich sehr weit unten in der Hierarchie befindet, da alle Elternelemente erweitert werden müssen. Zudem funktioniert die Suche nur dann effizient, wenn der Weg offensichtlich oder bekannt ist, da der Benutzer nicht sehen kann, was sich in den nicht erweiterten Pfaden befindet.

2.4.4 Tree-Map

Zeigt ein Dendrogramm die Hierarchie *von der Seite*, so entspricht eine Tree-Map einer Ansicht *von oben*. Jede Gruppe und Untergruppe ist von einer Linie umrandet, wodurch die Hierarchie sichtbar wird. Dabei kann diese Umrandung beliebige Formen haben, Abb. 2.12 zeigt ein Beispiel mit Rechtecken. Im Vergleich zum Dendrogramm ist jedoch schwerer ersichtlich, wann Gruppen verschmolzen wurden. Außerdem kann diese Visualisierung schnell zu schlecht lesbaren Layouts führen, da bei einer großen Zahl von Elementen die sie repräsentierenden Flächen sehr klein werden.

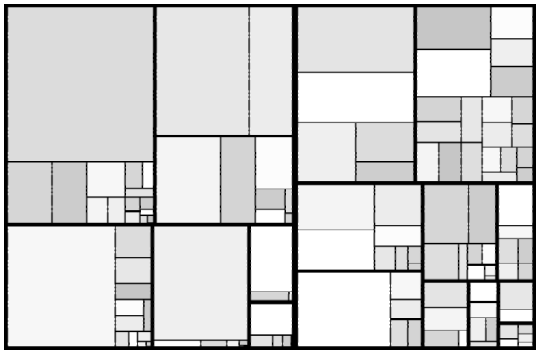


Abbildung 2.12: Schematische Darstellung einer Tree-Map mit Rechteck-basiertem Layout. Jedes der Rechtecke entspricht einer Gruppe, die ausgefüllten Flächen repräsentieren die geclusterten Elemente. Die Stärke der Linien sowie die Anordnung und Größe der Rechtecke zeigen die Zugehörigkeit von Elementen und Gruppen zu ihren übergeordneten Gruppen. Das Bild wurde übernommen von [39] und zugeschnitten.

2.4.5 Multidimensionale Skalierung

Bei einer multidimensionalen Skalierung [19] werden Ähnlichkeiten visualisiert, indem die zu zeigenden Daten auf einem meist zweidimensionalen Bereich so angeordnet werden, dass die Abstände aller Paare ihren Ähnlichkeiten entsprechen, was dazu führt, dass sich ähnlichere Elemente näher beieinander befinden als unterschiedlichere. Ein Beispiel für diese Visualisierung ist in Abb. 2.13 gezeigt.

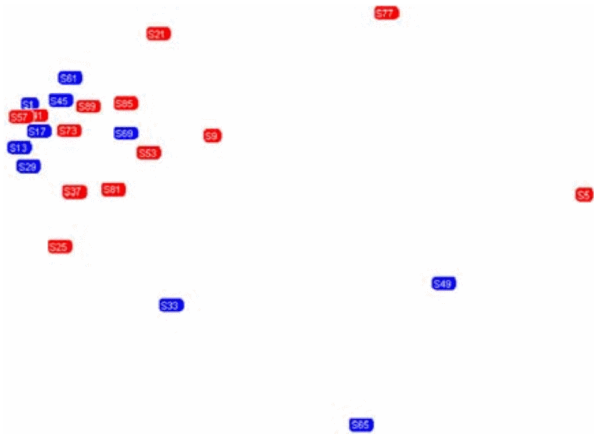


Abbildung 2.13: Visualisierung von Ähnlichkeiten mit multidimensionaler Skalierung. Die Elemente befinden sich umso näher beieinander, je ähnlicher sie sich sind. Der Wert einer gewählten Variable wird zusätzlich durch Farben visualisiert. Das Bild stammt aus [46] und wurde zugeschnitten.

3 Aufgabe und Lösungsansatz

Dieses Kapitel stellt zunächst das Szenario (Abschnitt 3.1) vor, in das diese Arbeit eingeordnet ist. Der zweite Abschnitt (3.2) erläutert die Aufgabenstellung, die durch den im letzten Abschnitt (3.3) gezeigten Ansatz gelöst werden soll.

3.1 Szenario

In letzter Zeit wurde Eye-Tracking eine immer beliebtere Methode zur Evaluation von Benutzerschnittstellen. In vielen Fällen ist eine Untersuchung der von Benutzern angewandten Strategien zur Lösung eines Problems hilfreich für eine weitere Optimierung der Darstellung. Ein gutes Beispiel dafür sind Visualisierungskonzepte, bei denen ein Anwender interaktiv in die gezeigte Ansicht eingreifen kann. Subjektive Bewertungen, Aufzeichnungen und sogenannte Think-Aloud-Protokolle sind nicht immer in der Lage, einen Eindruck davon zu liefern, womit die Person in einem bestimmten Moment gedanklich beschäftigt war. Unterbewusste Vorgänge können nur indirekt beobachtet werden. Eye-Tracking bietet hier eine Alternative, da davon ausgegangen werden kann, dass ein momentan von den Augen fixiertes Objekt mit hoher Wahrscheinlichkeit jenes ist, mit dem das Gehirn in diesem Augenblick beschäftigt ist.

Ein bestehendes Problem stellt jedoch die Auswertung der aufgenommenen Augenbewegungen dar. Eine Reihe von Analyse- und Visualisierungswerkzeugen nimmt sich unter Verfolgung verschiedener Ansätze dieser Problematik an (siehe Kapitel 4). Gerade beim Vergleich von Scanpaths auf Ähnlichkeiten im Blickverhalten und der damit vermuteten Strategie von Probanden wurde noch keine optimale Lösung gefunden.

3.2 Aufgabenstellung

Das Ziel dieser Forschungsarbeit ist der Vergleich und die Bewertung mehrerer Konzepte zur Gruppierung von Scanpaths anhand der Ähnlichkeit der von den Probanden verwendeten Suchstrategien. Zunächst soll daher eine Recherche zu vorhandenen Metriken zur Berechnung der Ähnlichkeit zwischen Scanpaths, beziehungsweise deren String-Repräsentationen, durchgeführt werden. Dazu findet außerdem eine Suche nach verwandten Arbeiten im Bereich Scanpath-Vergleich und -Visualisierung statt. Die Ergebnisse dieser Recherche werden in den Kapiteln 2 und 4 beschrieben. Anschließend soll ein Konzept erarbeitet werden, das auf den Ergebnissen der Recherche sowie eigenen Anpassungen und Erweiterungen beruht. Dieses Konzept soll in einem Prototypen implementiert und anhand mehrerer Szenarien evaluiert werden.

3.3 Lösungsansatz

Die bei der Recherche gefundenen Vergleichstechniken sollen auf ihre jeweiligen Vor- und Nachteile hin untersucht werden, um ihre Eignung für den Vergleich von Suchstrategien anhand von Scanpaths zu bewerten. Außerdem werden Überlegungen zu möglichen Anpassungen und Erweiterungen getroffen. Dazu gehört unter anderem eine Einbeziehung der in den AOIs implizit enthaltenen Informationen über deren Ähnlichkeit zueinander. Es soll weiterhin ein Konzept für eine geeignete Aufbereitung und Visualisierung der entstehenden Vergleichsdaten entwickelt werden, um eine für den Analysten leicht verständliche Repräsentation der Ergebnisse zu schaffen. In einem Experiment soll schließlich getestet werden, wie leistungsfähig sich die ausgewählten Metriken in unterschiedlichen Parameter-Konfigurationen bei der Verarbeitung von Testdaten zeigen. Die ungefähre Vorgehensweise bei diesem Experiment ist in Abb. 3.1 dargestellt.

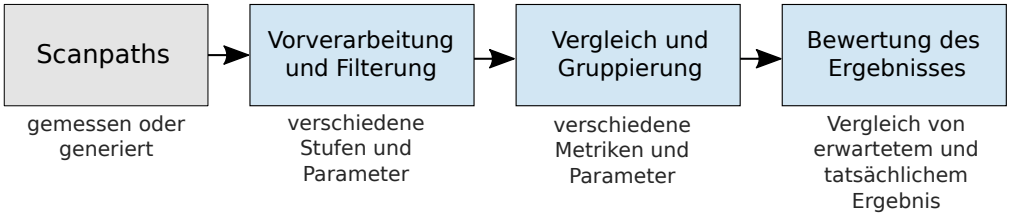


Abbildung 3.1: Lösungsansatz für die Bewertung der Metriken. Die verwendeten Scanpaths können aus einer Studie stammen oder generiert sein. Wichtig ist jedoch, dass die ungefähre Ähnlichkeit unter ihnen bekannt ist. Sie werden vorverarbeitet und verglichen, wobei verschiedene Techniken und Parameter zum Einsatz kommen. Je nach erwartetem Ergebnis wird dann eine Anzahl von Gruppen nach Ähnlichkeit gebildet. Das Ergebnis wird bewertet, indem die korrekte Gruppierung überprüft wird.

4 Existierende Arbeiten

In diesem Kapitel werden verwandte Ansätze und Implementierungen vorgestellt, die sich mit der Analyse und dem Vergleich von Scanpaths beschäftigen. Während Programme wie ScanMatch und EyePatterns mit AOIs arbeiten, verfolgt MultiMatch einen vektor-basierten Ansatz, mit dem Scanpaths in verschiedener Hinsicht verglichen werden können. Der mit Parallel-Scanpaths umgesetzte Ansatz versucht sich hingegen an einer Vereinfachung des Scanpath-Vergleichs mittels Methoden aus dem Bereich *Visual Analytics*.

4.1 ScanMatch

Mit ScanMatch [4] wurde eine Toolbox implementiert, mit deren Hilfe Scanpaths auf ihre Ähnlichkeit hin untersucht werden können. Die folgenden Abschnitte erläutern die Vorverarbeitung der Scanpaths und zum Vergleich verwendete Verfahren. Außerdem wird die Evaluation zur Beurteilung des Verfahrens beschrieben. Abschließend werden die Stärken und Schwächen dieses Ansatzes in einem Fazit zusammengefasst.

4.1.1 Vorverarbeitung

Vor der Durchführung des Vergleichs wurden die Scanpaths durch Verwendung einer zeitlichen Quantisierung so vorverarbeitet, dass auch die Dauer von Fixationen in die Berechnung einfließt. Weiterhin wurde ein Schema zur Repräsentation der AOIs im Scanpath angewandt.

Um zeitliche Information in den Scanpath einzubringen, wurde ein sogenanntes *Temporal Binning* eingeführt. Dabei wird das dem AOI-Aufenthalt zugeordnete Symbol entsprechend der Aufenthaltszeit wiederholt in den Scanpath-String eingefügt. Die Zeitspanne, nach welcher ein weiteres Zeichen hinzugefügt wird, wurde auf 50 Millisekunden festgelegt, um die übliche Fixationsdauer von 100 bis 1000 Millisekunden nach dem Abtasttheorem von Nyquist und Shannon [32] korrekt quantisieren zu können (siehe Abb. 4.1).

Da bei der Nutzung von einzelnen Buchstaben für AOI-Kennungen nur eine kleine Zahl von AOIs möglich ist, wurde eine Kodierung der AOIs mit zwei Buchstaben verwendet. Dabei wurde zur Verbesserung der Lesbarkeit der jeweils erste Buchstabe kleingeschrieben.

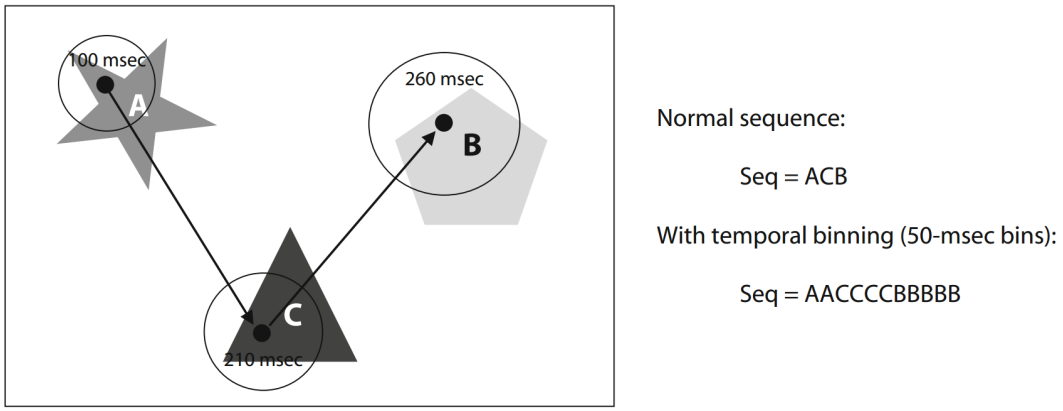


Abbildung 4.1: Ein Beispiel für einen Scanpath und dessen String-Repräsentation. Ohne Temporal Binning würden die Fixationen beziehungsweise AOI-Aufenthalte den Scanpath-String *ACB* ergeben. Nach einem Binning mit einem Zeichen stellvertretend für jeden Zeitabschnitt von 50 Millisekunden entsteht ein längerer String mit quantisierten Zeitinformationen [4].

4.1.2 Verfahren

Der Vergleich der Scanpaths basiert auf dem Algorithmus von Needleman und Wunsch [30], der in Abschnitt 2.3.4 vorgestellt wurde. Dieser Abschnitt erläutert die Wahl der Parameter für den Algorithmus sowie eine Normalisierung anhand dieser Parameter und der Länge der Scanpath-Strings.

Die *Ersetzungsmatrix* für den Algorithmus von Needleman und Wunsch wurde anhand euklidischer Distanzen angelegt. Zwei AOIs gelten als ähnlicher, wenn sie sich räumlich näher beieinander befinden. Es sind allerdings auch andere Metriken denkbar, etwa die Ähnlichkeit der Farbe oder semantische Beziehungen. Entsprechend der Ähnlichkeit der AOIs wird dann die Ersetzungsmatrix für den Algorithmus aufgestellt. Um kleinere Punktzahlen für nur schwach verwandte AOIs zu erhalten, wurde ein Grenzwert ermittelt, ab dem der entsprechende Wert in der Ersetzungsmatrix negativ werden soll. Dafür wurde die doppelte Standardabweichung aller Sakkadenamplituden genommen.

Ein zusätzlich zu der Ersetzungsmatrix benötigter Parameter ist die *Gap Penalty*, die eine Zuordnung eines Zeichens zu einer Lücke mit negativen Punkten bestraft. Bei ScanMatch wurde aufgrund des Thresholds, der bereits für negative Punkte bei unterschiedlichen AOIs sorgt, eine Gap Penalty von null gewählt.

Das Ergebnis des Algorithmus ist für längere Zeichenketten bei gleicher Ähnlichkeit höher als für kurze. Daher wird eine Normalisierung durchgeführt, sodass bei einem Vergleich von zwei identischen Strings ein Ergebnis von 1 entsteht.

$$\text{Normalisiertes Ergebnis} = \frac{\text{Ergebnis}}{\text{Maximum der Ersetzungsmatrix} \times \text{Länge des längeren Strings}} \tag{4.1}$$

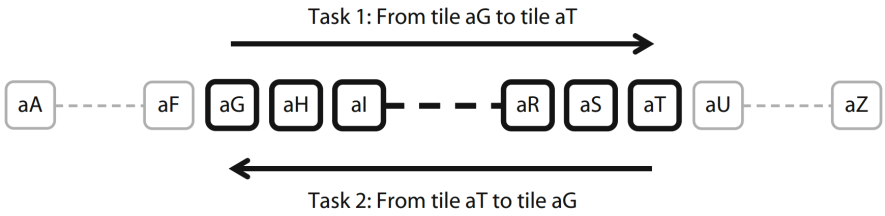


Abbildung 4.2: Erzeugung von Testdaten bei ScanMatch. Eine Reihe von AOIs wird entweder von links nach rechts oder umgekehrt fixiert [4].

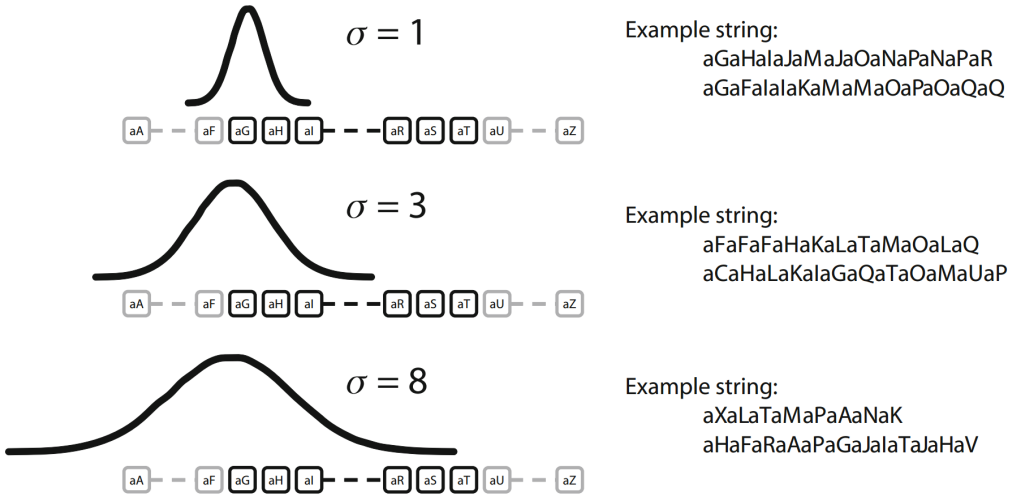


Abbildung 4.3: Erzeugung von Testdaten bei ScanMatch. Um verschiedene Scanpaths zu erzeugen, werden Fixationen zufällig normalverteilt gesetzt. Der Erwartungswert liegt dabei auf der nach der Aufgabe eigentlich angepeilten AOI. Je nach Wert des Parameters σ ergeben sich mehr oder weniger unterschiedliche Strings [4].

4.1.3 Evaluation

ScanMatch wurde in drei Versuchen evaluiert. Beim ersten Versuch kamen künstlich erstellte Testdaten zum Einsatz, um den Ansatz zu testen und mit der etablierten Levenshtein-Distanz zu vergleichen. Dazu wurden 26 AOIs in einer Reihe angeordnet, wie auf Abb. 4.2 zu sehen. Als hypothetische Aufgabe wurde das Betrachten aller AOIs zwischen der siebten (aG) und siebt letzten (aT) gewählt. Dies sollte in der richtigen Reihenfolge, bei Aufgabe 1 von links nach rechts und bei Aufgabe 2 umgekehrt, und ohne Auslassung geschehen. Fixationen wurden dann zufällig normalverteilt gesetzt, wobei der Erwartungswert der Normalverteilung bei jeder Fixation je nach der gewünschten Scanpath-Richtung eine AOI weiter nach links beziehungsweise rechts versetzt wurde. Diese Verteilung der Fixationen und die dadurch erreichte Erzeugung der Scanpaths ist in Abb. 4.3 zu sehen. Der einzige Parameter, der bei jedem erzeugten Scanpath variiert wurde, ist die Standardabweichung der Normalverteilung. Je größer sie ist, desto unterschiedlicher werden die erzeugten Scanpaths.

Auf diese Weise wurden je hundert Scanpaths von links nach rechts und rechts nach links generiert, wobei die Standardabweichung zwischen 0 und 14 in Schritten von 0,5 verändert wurde. Diese Scan-

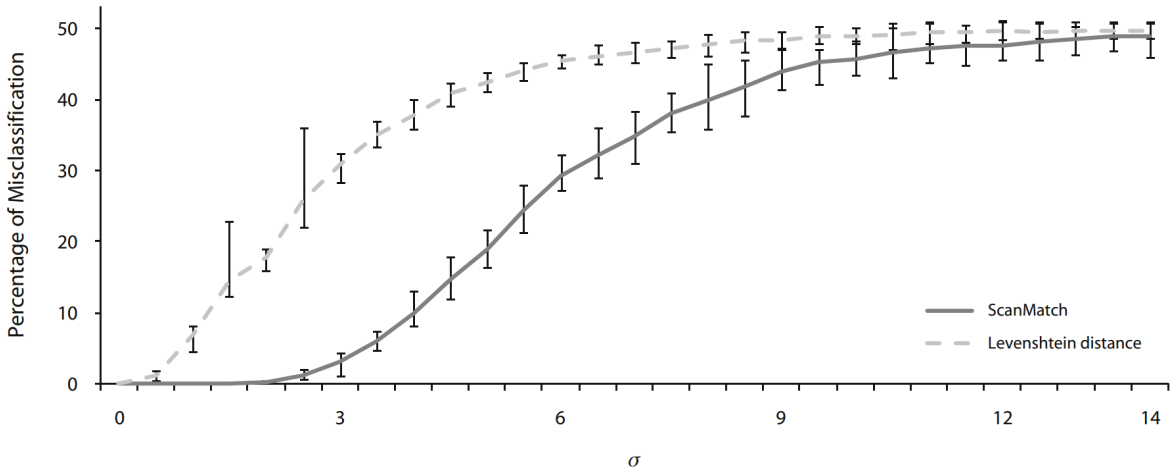


Abbildung 4.4: Vergleich von ScanMatch und Levenshtein-Distanz im ersten Versuch mit automatisch generierten Testdaten. ScanMatch liefert durchgehend bessere Resultate, insbesondere bei einer Standardabweichung $\sigma < 9$ [4].

paths wurden dann jeweils mit dem Levenshtein-Algorithmus und dem von Needleman und Wunsch verglichen. Erwartungsgemäß sollten Scanpaths, die aus der selben Aufgabe resultierten, als ähnlich bewertet werden. Umgekehrt wurde erwartet, dass nach verschiedenen Aufgaben erzeugte Pfade vom Algorithmus als zueinander unterschiedlich bewertet werden. Durch *k-Means-Clustering* wurde versucht, die Scanpaths nach ihren Aufgaben zu gruppieren und der Prozentsatz der Fehleinordnungen wurde als Maß für die Korrektheit des Verfahrens genommen. Abb. 4.4 zeigt diese Prozentsätze vergleichsweise für Levenshtein und ScanMatch, wobei letzteres vor allem bei schwach verrauschten Daten, aber auch bei größerem σ, deutlich bessere Werte erzielt. Bis zu einer Standardabweichung von 2,5 ordnet ScanMatch alle Scanpaths fehlerfrei zu.

Beim zweiten Versuch wurden Probanden angewiesen, Zahlen von eins bis neun entweder in Grün oder Rot und entweder auf- oder absteigend auf dem Stimulus zu fixieren. Dies führte erneut zu zwei verschiedenen Gruppen von Scanpaths. Mit einer Unterteilung in 12×8 gitterförmig angeordneten AOIs und dem Anlegen einer Ersetzungsmatrix basierend auf euklidischen Abständen konnte ScanMatch alle Aufzeichnungen korrekt zuordnen.

Der dritte Versuch verlangte vom Probanden, zwischen mehreren Vorkommen des Buchstaben *L*, die verschieden gedreht waren, ein *T* zu finden und dessen Orientierung anzugeben. Dabei kamen rote und grüne Buchstaben vor. Für die Ersetzungsmatrix wurde in diesem Versuch *Farbe* als Ähnlichkeitsmaß verwendet, die Werte waren abhängig von der Farbe des Ziels, also der des gesuchten *T*. Auch hier war eine gute Zuordnung der Pfade zu den intendierten Gruppen entsprechend des Suchzieles möglich.

4.1.4 Fazit

Die beiden Vorteile des in ScanMatch gezeigten Ansatzes sind die Einbeziehung zeitlicher Informationen durch Temporal Binning und die Abhängigkeit der Ersetzungskosten von der Ähnlichkeit der

AOIs. Letztere kann unter anderem anhand von euklidischer Distanz, Farbe oder Semantik bestimmt werden.

Das eingeführte Temporal-Binning ermöglicht zwar die Einbeziehung der Fixationsdauer, führt aber zu einer zeitlichen Quantisierung. So könnte eine Fixation, die 130 Millisekunden dauerte, bei einer Binning-Zeitspanne von 50 Millisekunden zu zwei Zeichen ab- oder zu drei Zeichen aufgerundet werden [7]. Die Ersetzungsmatrix ist über die Zeit der Berechnung konstant und kann daher beispielsweise keinen Lerneffekt oder AOIs mit zeitabhängiger Ähnlichkeit berücksichtigen. Außerdem ergeben sich die bei der Nutzung von AOIs auftretenden Probleme, etwa dass Fixationen, die innerhalb einer AOI weit auseinander liegen, als ähnlicher betrachtet werden, als solche, die sich beidseitig einer AOI-Grenze nahe zusammen befinden.

4.2 MultiMatch

Der in MultiMatch [7, 17] umgesetzte Ansatz basiert auf Vektoren, die Sakkaden exakt repräsentieren. Scanpaths werden in mehreren Kriterien verglichen, wobei räumliche und zeitliche Informationen genutzt werden. Ziel war vor allem eine Erkennung von ähnlichen Formen, auch wenn diese verschieden skaliert oder zueinander verschoben sind. Ein Vergleich mit ScanMatch wurde vorgenommen, um die Leistungsfähigkeit der Implementierung zu zeigen. Auch für diesen Ansatz werden die Vorverarbeitung, das Vergleichsverfahren und die Evaluation in je einem Abschnitt behandelt und danach ein Fazit gezogen.

4.2.1 Vorverarbeitung

Um beim eigentlichen Vergleich ein besseres Ergebnis zu erhalten, insbesondere bei kleineren Abweichungen aufgrund von Rauschen, werden die Scanpaths zunächst vorverarbeitet. Sie werden dazu durch eine Glättung vereinfacht, wie Abb. 4.5 zeigt. Dabei werden kleinere Gruppen von Sakkaden unter Verwendung von Grenzwerten für Amplituden und Winkelabweichungen zu einer einzelnen Sakkade zusammengefasst, ähnlich wie bei der Berechnung von Fixationen aus rohen Eye-Tracking-Daten. Um wichtige Fixationen und Sakkaden beizubehalten, wird vorgeschlagen, diejenigen von der Zusammenfassung auszunehmen, deren Dauer einen festgelegten Grenzwert überschreitet.

4.2.2 Verfahren

Als erster Schritt des eigentlichen Vergleichs wird eine zeitliche Zuordnung der Vektoren beider Scanpaths zueinander durchgeführt. Dies geschieht mit Hilfe einer Vergleichsmatrix mit den jeweiligen Unterschieden der Vektoren, wobei die Differenz der Vektorlängen in visuellen Grad genutzt wird (siehe Abb. 4.6). Diese Matrix wird als Graph aufgefasst und durch diesen mittels des *Dijkstra-Algorithmus* [8] der kürzeste Pfad gefunden. Die Vektoren werden dann entsprechend diesem Pfad einander zugeordnet.

Die Scanpaths können nun verglichen werden, indem die Unterschiede in den verschiedenen Dimensionen betrachtet werden: Für den Vergleich der Form werden die Vektoren subtrahiert und zur

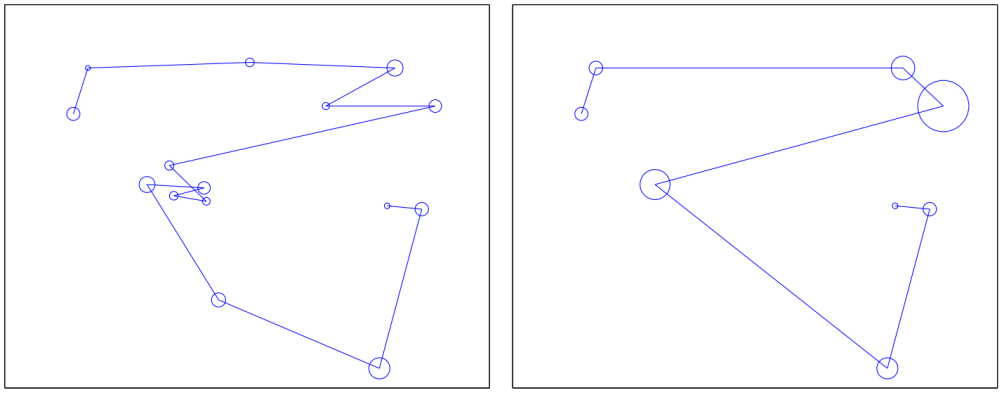
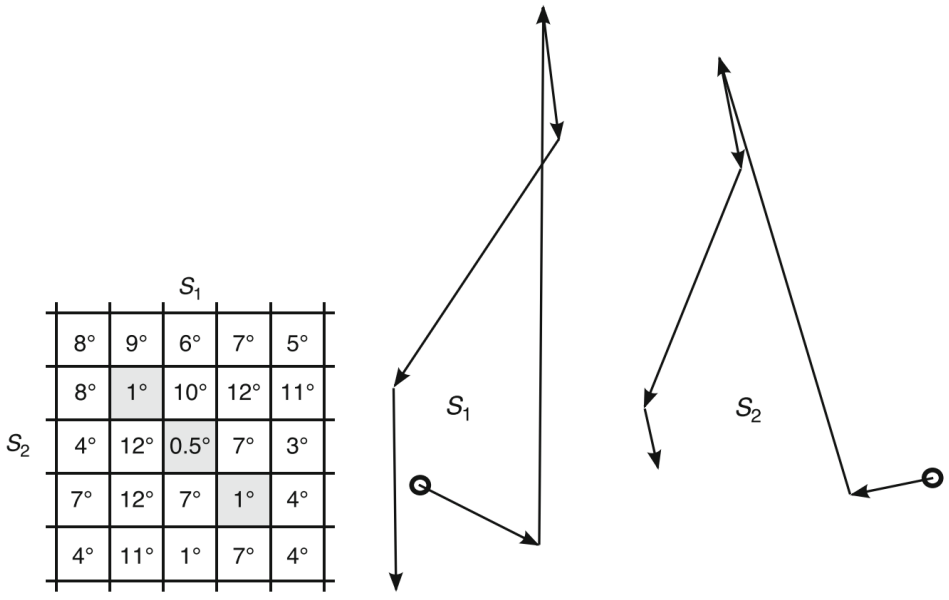


Abbildung 4.5: Ein Beispiel für die Vereinfachung der Scanpaths bei MultiMatch. Links ist der ursprüngliche, rechts der vereinfachte Pfad zu sehen. Unter Nutzung von Grenzwerten findet bei der Vereinfachung eine Zusammenfassung von Vektoren mit ähnlicher Richtung oder solchen mit geringer Länge an ähnlichen Positionen statt [17].



(a) Comparison matrix. (b) Scanpaths.

Abbildung 4.6: Die Vergleichsmatrix zweier Scanpaths. Die Werte entsprechen den jeweiligen Differenzen der Vektorlängen in visuellen Grad. Kleine Werte stehen für ähnliche Vektoren [15].

doppelten Bildschirmdiagonale normalisiert. Die Längen werden ebenfalls durch ihre mathematische Differenz verglichen und wie der Abstand der Positionen mit der Bildschirmdiagonalen normalisiert. Die Differenz in der Richtung der Vektoren wird durch π normalisiert, der Unterschied zwischen zwei Fixationsdauern durch die größere der beiden. So ergibt sich für jede Dimension ein Wert zwischen 0 und 1, wobei 1 für identisch steht.

4.2.3 Evaluation

MultiMatch wurde mit ScanMatch in seiner Standardkonfiguration in zwei Versuchen verglichen. Der erste sollte die Anfälligkeit für Rauschen in generierten Daten zeigen. Der zweite sollte Vergleichsdaten zur Leistungsfähigkeit in Hinsicht auf die bekannten Probleme mit AOIs und verschieden skalierte Scanpaths liefern und verwendete aufgezeichnete Eye-Tracking-Daten von echten Probanden.

Die Vorgehensweise beim ersten Versuch basierte ähnlich der Evaluation von ScanMatch auf zufällig generierten Scanpaths. Allerdings wurden in diesem Fall zwei verschiedene Pfade erzeugt und von diesen wiederum zufällig veränderte Varianten erstellt. Dabei wurde jede Fixation in ihrer Position um einen Wert geändert, der zufällig normalverteilt ermittelt wurde. Je nach gewünschter Unterschiedlichkeit zum originalen Pfad wurde die Standardabweichung zwischen 10 und 90 Prozent der Breite des Stimulus gewählt. Als Ergebnis entstanden zwei Gruppen von Scanpaths mit höherer Ähnlichkeit innerhalb der Gruppen als zwischen ihnen. In diese Gruppen sollen die Pfade später eingeordnet werden. Ein Beispiel für auf diese Weise gewonnene Scanpaths ist in Abb. 4.7 gezeigt. Sowohl MultiMatch als auch ScanMatch konnten die generierten Pfade nach ihrer Ähnlichkeit einordnen, auch wenn bei stark verrauschten Daten die Qualität der Ergebnisse abnahm.

Beim zweiten Versuch wurden, wie bei einer Eye-Tracker-Kalibrierung, nacheinander mehrere Punkte angezeigt, die vom Probanden fixiert werden sollten. Dabei wurden Punkte aus Pfaden verwendet, die jeweils in bestimmten Dimensionen ähnlich zueinander sind, siehe Abb. 4.8. Da dieser Versuch speziell auf die Stärken von MultiMatch fokussiert war, ist es nicht überraschend, dass ScanMatch hier in seiner Leistung übertroffen wurde, was vor allem an der Nutzung der AOIs liegt.

4.2.4 Fazit

Durch Verzicht auf AOIs umgeht MultiMatch die bereits erwähnten Probleme dieser Quantisierung. Im Gegensatz zu ScanMatch entfällt außerdem auch die zeitliche Quantisierung durch das dort verwendete Temporal Binning. Der Vergleich von Scanpaths unterschiedlicher Länge wird verbessert. Dies ist für viele vorhandene Metriken ein großes Problem gewesen. Ein erheblicher Vorteil dieses Ansatzes ist zudem, dass Scanpaths anhand mehrerer Dimensionen verglichen werden, wodurch beispielsweise zwei Pfade, die nur in manchen Dimensionen abweichen, leichter als ähnlich erkennbar sind. Das ist etwa in Studien nützlich, in denen manche der Kriterien besonders wichtig oder vernachlässigbar sind, etwa wenn nur die Form zweier Scanpaths ähnlich sein soll. Anders formuliert zeigt MultiMatch nicht nur die Ähnlichkeit an, sondern auch worin diese Ähnlichkeit besteht. Durch das Fehlen von AOIs entstehen jedoch auch Nachteile. So können beim Vergleich zweier Scanpaths keine Informationen über die Semantik der betrachteten Bereiche miteinbezogen werden, wie es bei ScanMatch etwa mit der Farbe der im Stimulus enthaltenen Symbole möglich war.

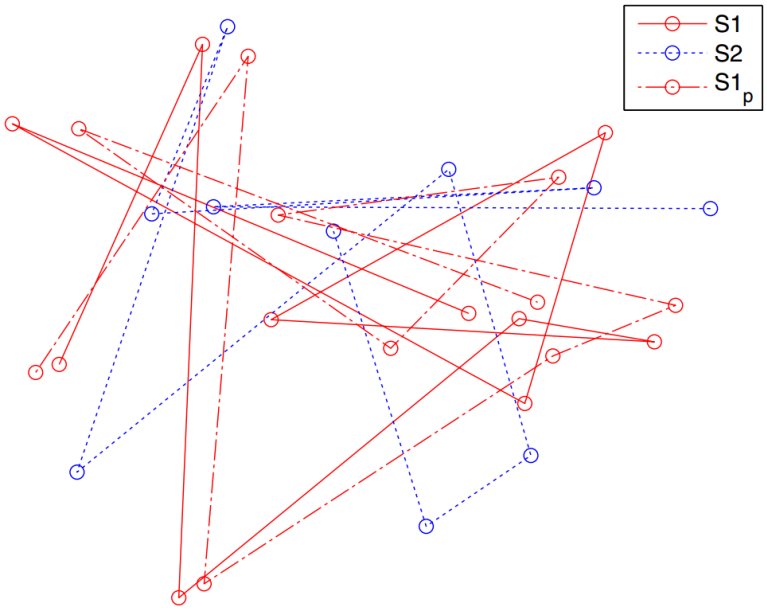


Abbildung 4.7: Zwei zufällig erzeugte Scanpaths ($S1$, $S2$) und eine Variante ($S1_p$), die aus $S1$ durch Hinzufügen von Rauschen entstand. Beim Vergleich der Pfade sollten $S1$ und $S1_p$ als zueinander ähnlicher erkannt werden als jeweils zu $S2$ [7].

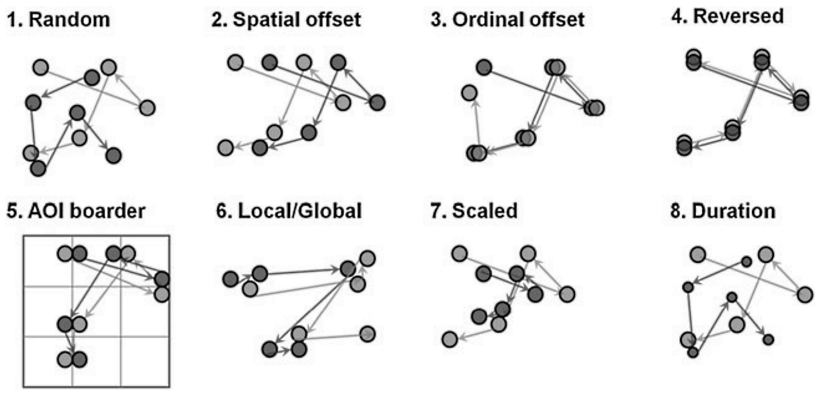


Abbildung 4.8: Beispiele für Paare von Scanpaths mit Ähnlichkeit in verschiedenen Vergleichsdimensionen [7].

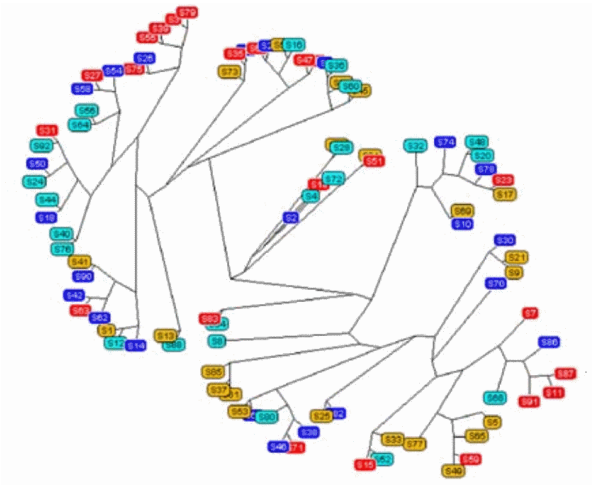


Abbildung 4.9: Visualisierung der Scanpath-Ähnlichkeit durch einen Baum, der ausgehend von der Mitte dargestellt ist. Die Farben stehen für Werte einer gewählten Variablen, in diesem Fall die an die Probanden gestellte Aufgabe [46] (Bild wurde zugeschnitten).

4.3 EyePatterns

Das Programm EyePatterns [46] bietet dem Benutzer verschiedene Möglichkeiten zum Vergleich von Scanpaths über einem Stimulus mit AOIs.

4.3.1 Verfahren

Als Vergleichsmetriken dienen die Algorithmen von Levenshtein, Needleman und Wunsch sowie der von Smith und Waterman, die in den Grundlagen dieser Arbeit vorgestellt wurden (siehe Abschnitte 2.3.4 und 2.3.5). Wie bei diesen Algorithmen üblich, wurden die Scanpaths durch Strings repräsentiert, die je ein Zeichen für jede Fixation enthalten. Um die Anzahl von Fixationen innerhalb der AOI-Aufenthalte außer Acht lassen zu können, wurde eine optionale Zusammenfassung von mehreren aufeinanderfolgenden identischen Buchstaben zu einem einzelnen Zeichen genutzt.

4.3.2 Visualisierung

Als Ergebnis wird eine Tabelle mit den Ähnlichkeitswerten aller Scanpath-Paare ausgegeben. Außerdem kann ein hierarchisches Clustering durchgeführt werden, dessen Resultat als Baum visualisiert wird, wie in Abb. 4.9 zu sehen. Auch eine Darstellung als Multidimensionale Skalierung ist verfügbar, siehe dazu Abschnitt 2.4.5 und Abb. 2.13.

Ein interessantes Feature ist die Erkennung von Mustern, beziehungsweise Teilsequenzen, die in einem oder mehreren Scanpaths wiederholt vorkommen. Muster aus zwei verschiedenen Scanpaths werden einander mit dem Algorithmus von Smith und Waterman zugeordnet. Wie diese Zuordnung aussehen kann, ist in Abb. 4.10 dargestellt. Solche Muster können darauf hindeuten, dass zwei Probanden

Abbildung 4.10: Visualisierung der einander zugeordneten Muster zweier Scanpaths bei EyePatterns [46].

eine ähnliche Strategie verwendeten, jedoch zu unterschiedlichen Zeiten. Mit Hilfe von regulären Ausdrücken können beliebige Muster in den Daten gesucht werden, eine Eingabemaske vereinfacht die Bedienung zusätzlich. Auf diese Weise ist eine Suche nach ungefähren Übereinstimmungen möglich.

4.4 eSeeTrack

Mit eSeeTrack [42] wurde ein Werkzeug entwickelt, das Muster aus Scanpaths extrahiert und visualisiert. Dazu werden zwei Zeitleisten und eine baumartige Visualisierung genutzt, wie Abb. 4.11 zeigt. Es werden sowohl statische als auch dynamische Stimuli unterstützt. Das Ziel ist der Vergleich von mehreren Gruppen von Versuchsteilnehmern.

Vor der Benutzung von eSeeTrack muss der Analyst zuerst AOIs festlegen und die Fixationen diesen zuordnen lassen. Bei dynamischen Daten ist eine manuelle Zuordnung nötig. Danach können die Teilnehmer in bis zu sechs Gruppen eingeteilt werden. Jede dieser Gruppen bekommt je eine zusammenfassende und eine detaillierte Zeitleiste. Diese sind durch Dreiecksmarken in gleichmäßige Abschnitte nach Probanden unterteilt. Die Fixationen werden innerhalb der Leisten als farbige Balken visualisiert, die Farbe steht dabei für die jeweils fixierte AOI. So zeichnen sich auf einen Blick Ähnlichkeiten in der Häufung einer bestimmten AOI bei mehreren Probanden ab.

Die Baumvisualisierungen aller Gruppen werden überschneidend dargestellt. Die Wurzel des Baums ist ein in der Detail-Zeitleiste ausgewähltes Tag. Sie kann links oder rechts vom Baum stehen. Im ersten Fall werden die nachfolgenden, im zweiten die vorangegangenen Blickmuster gezeigt. So kann herausgefunden werden, was ein Versuchsteilnehmer vor oder nach einem gewählten Objekt betrachtet hat. Die Textgröße entspricht dabei der relativen Betrachtungshäufigkeit, der am häufigsten vorkommende Pfad wird oben abgebildet.

Im Gegensatz zu den anderen bereits vorgestellten Arbeiten wird bei dieser kein algorithmischer Vergleich der Scanpaths durchgeführt. Die verwendete Visualisierung kann bei längeren zu zeigenden Mustern überladen wirken. Allerdings erlauben interaktive Funktionen und die beiden Zeitleisten eine visuelle Analyse der Daten.

Abbildung 4.11: Die Benutzeroberfläche von eSeeTrack. Oben befinden sich die Zeitleiste (a) und die Detail-Zeitleiste (b), mit denen sich der Benutzer einen Überblick über die Fixationssequenzen verschaffen kann. Im unteren Bereich kann mit Hilfe einer Baumansicht (c) herausgefunden werden, welche Sequenzen unmittelbar vor oder nach einer ausgewählten Fixation vorkamen. Damit können häufig vorkommende Muster in den Scanpaths gefunden werden. Rechts stehen Kontrollelemente (d) zur Auswahl der Daten und Parameter zur Verfügung [42].

4.5 Parallel-Scanpath-Visualisierung

Wie bei eSeeTrack wurde mit der Parallel-Scanpath-Visualisierung [35] eine Möglichkeit geschaffen, das Blickverhalten von Probanden zu vergleichen.

4.5.1 Verfahren

Der Ansatz für eine übersichtliche Visualisierung ist die Darstellung von AOIs als parallele Achsen. Auf diesen Achsen werden die Daten nach der Zeit ihres Auftretens eingezeichnet. Punkte auf diesen Achsen stehen für Fixationen oder AOI-Aufenthalte. Die Punkte werden durch Linien verbunden, die Transitionen zwischen AOIs repräsentieren. Am Rand der Visualisierung befindet sich eine Zeitachse, welche die Richtung und Koordinaten des zeitlichen Verlaufs anzeigt.

Auf diesem Konzept basieren drei verschiedene Arten von Visualisierungen. Ein *Gaze Duration Sequence Diagram* bildet die aufeinanderfolgenden AOI-Aufenthalte entsprechend ihrer Dauer ab. Vertikale Linienabschnitte stellen AOI-Aufenthalte dar, während horizontale Verbindungen zwischen den Achsen für Übergänge zwischen zwei AOIs stehen. Einzelne Fixationen werden durch ein *Fixation*

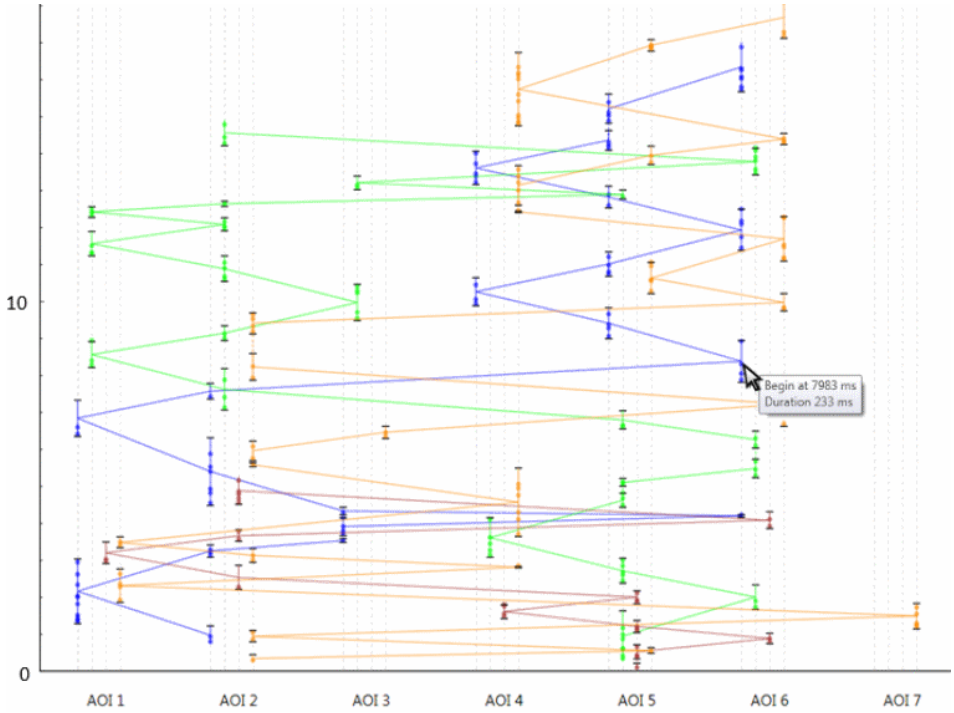


Abbildung 4.12: Beispiel für ein *Fixation Point Diagram*. Hier wurden die Scanpaths von vier Probanden (blau, grün, braun und violett) gleichzeitig dargestellt. Für jeden davon steht eine eigene Achse pro AOI zur Verfügung. Die Punkte entsprechen den aufgezeichneten Fixationen, die entsprechend ihres Zeitpunktes und der fixierten AOI in das Diagramm eingezeichnet wurden. Unmittelbar übereinander liegende Punkte gehören zu einem gemeinsamen AOI-Aufenthalt. Die Aufenthalte sind durch Linien verbunden, um die zeitliche Abfolge sichtbar zu machen [35].

Point Diagram dargestellt (siehe Abb. 4.12). Ein *Gaze Duration Distribution Diagram* zeigt nur die zeitlichen Mittelpunkte der AOI-Aufenthalte, nicht ihre Dauer.

4.5.2 Verbesserung der Darstellung durch Gruppierung

Um die Lesbarkeit der Parallel-Scanpath-Visualisierung bei einer größeren Anzahl von Scanpaths zu verbessern, wurde von Raschke et al. [36] eine Gruppierung ähnlicher Pfade eingeführt. Die Ähnlichkeit wurde dabei mittels Levenshtein-Distanz bestimmt. Die Scanpaths einer Gruppe werden in der Visualisierung mit einer ähnlichen Farbe markiert oder nur noch durch einen Pfad repräsentiert. Dadurch wird der Überblick über die Daten verbessert. Abb. 4.13 zeigt einen Vergleich der Parallel-Scanpath-Visualisierung ohne und mit Gruppierung. Es wird deutlich, dass der ursprüngliche Ansatz sehr schlecht mit einer höheren Anzahl an Scanpaths skaliert, während eine Zusammenfassung ähnlicher Pfade zu einem Repräsentanten zwar Daten ausblendet, dafür aber für eine bessere Übersichtlichkeit der Visualisierung sorgt.

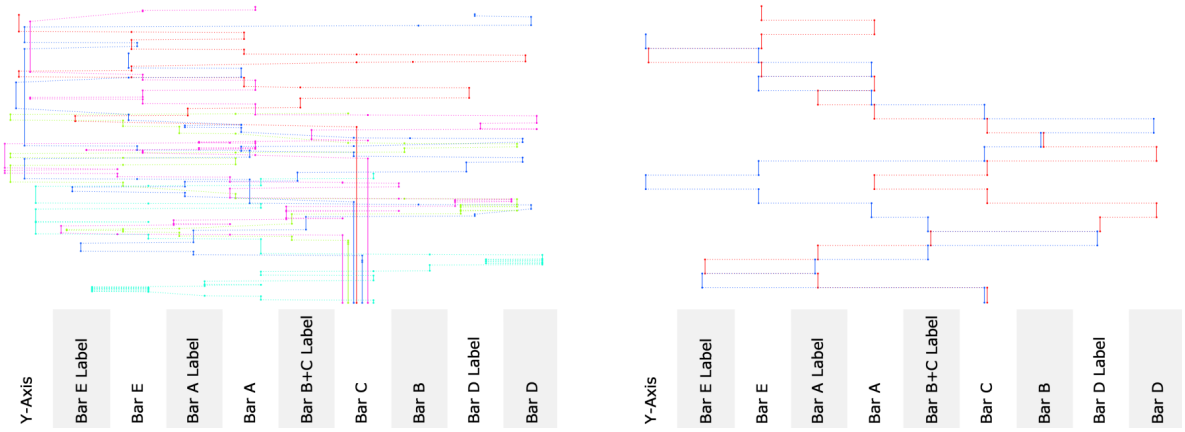


Abbildung 4.13: Vergleich von Visualisierungen vor (links) und nach (rechts) Gruppierung ähnlicher Scanpaths. Durch Clustering mit einem Grenzwert von 0,73 für die Ähnlichkeit der Cluster wurden zwei Gruppen erzeugt, die hier von jeweils einem Pfad repräsentiert werden. Außerdem wurden bei der rechten Visualisierung die einzelnen Fixationen sowie die Dauer der AOI-Aufenthalte nicht in die Visualisierung einbezogen. Dadurch werden alle Aufenthalte als vertikale Linien gleicher Länge dargestellt. Diese Maßnahmen verbessern die Lesbarkeit der Visualisierung und vereinfachen den Vergleich der Gruppen von Scanpaths [36].

Da im Allgemeinen keine feste Anzahl von Gruppen bekannt ist, wird die Gruppierung mittels hierarchischem Clustering (siehe Abschnitt 2.4.1) vorgenommen. Als Maß für die Ähnlichkeit zweier Scanpaths wird die Levenshtein-Distanz verwendet, die optional auch nur für einen Teilabschnitt der Pfade berechnet werden kann. Falls nur ein Scanpath als Repräsentant einer Gruppe dienen soll, wird derjenige gewählt, dessen Summe der Abstände zu allen anderen Pfaden in seiner Gruppe minimal ist.

Bei der Parallel-Scanpath-Visualisierung wird der Benutzer durch eine Veranschaulichung und optionaler Gruppierung der Pfade bei der Analyse der Daten unterstützt. Je nach Interesse können verschiedene Daten gezeigt oder verborgen werden, wie etwa die Dauer der Fixationen. Eine Darstellung der Ähnlichkeitshierarchie findet allerdings nicht statt.

5 Konzept

In diesem Kapitel wird detailliert auf das erarbeitete Konzept dieser Arbeit eingegangen. Die folgenden Abschnitte wägen Vor- und Nachteile der vorhandenen Metriken zum Vergleich von Scanpaths und Strings ab, zeigen Anpassungen und Erweiterungen, die vorgenommen werden können, und behandeln eine mögliche Repräsentation der zu vergleichenden Scanpaths. Danach wird auf eine Ordnung und Darstellung der Ergebnisse eines Scanpath-Vergleichs durch Clustering und eine passende Visualisierung eingegangen. Abschließend wird ein Konzept für die Evaluation der Leistungsfähigkeit der verschiedenen Metriken vorgestellt.

5.1 Überblick

Dieser Abschnitt soll einen Überblick über das anschließend ausführlich vorgestellte Konzept vermitteln. Alle Schritte sind zudem in Abb. 5.1 schematisch dargestellt.

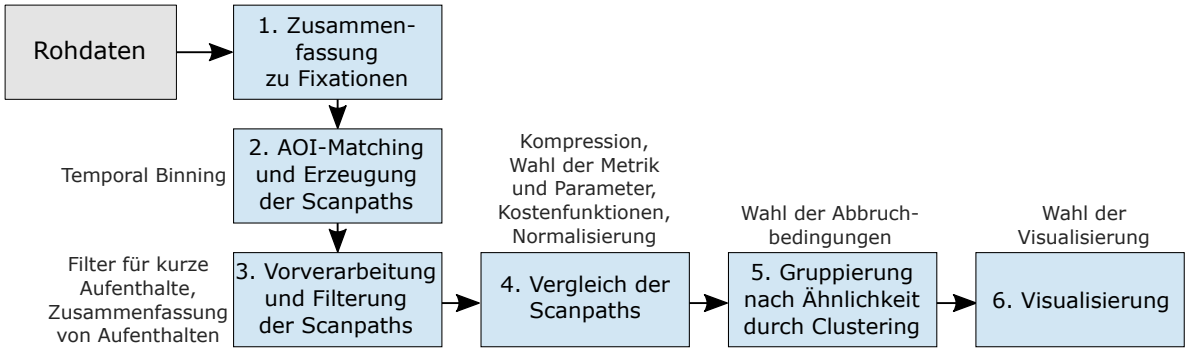


Abbildung 5.1: Überblick über das Konzept dieser Arbeit. Die vom Eye-Tracker aufgenommenen Rohdaten werden in den Schritten 1 bis 3 erst zu Fixationen und dann zu Scanpaths zusammengefasst. Dazu werden nacheinander alle Fixationen eines Probanden der sie enthaltenden AOI zugeordnet, und deren Kennung an den Scanpath angefügt. Hierbei werden bei Temporal Binning mehrere Zeichen proportional zur Dauer der Fixation hinzu genommen. Im 3. Schritt werden optional Sequenzen identischer Zeichen zu einem zusammengefasst oder diejenigen unter einer beliebigen Länge herausgefiltert. Anschließend werden die Scanpaths paarweise verglichen (Schritt 4), wobei eine Metrik und ihre Parameter gewählt werden können. Nach dem Vergleich werden die Pfade mittels Clustering gruppiert (Schritt 5). Zuletzt werden im 6. Schritt die Ergebnisse des Vergleichs und die erzeugten Gruppen visualisiert.

Das Ziel dieser Arbeit ist die Bestimmung der Ähnlichkeiten zwischen den Suchstrategien mehrerer Studienteilnehmer. Die Strategien werden anhand der aus den aufgezeichneten Daten gewonnenen Scanpaths verglichen. Nach dem Vergleich, bei dem verschiedene Metriken zum Einsatz kommen können, werden die Scanpaths gruppiert und die Gruppenhierarchie veranschaulicht. In den folgenden Absätzen werden die dafür nötigen Schritte aufgeführt, die im restlichen Kapitel ausführlich behandelt werden.

Nach der Durchführung einer Eye-Tracking Studie liegen zunächst nur die vom Eye-Tracker aufgenommenen Rohdaten vor. Diese enthalten für jeden Zeitpunkt, zu dem ein Sample aufgenommen wurde, die Positionen der Blicke der Probanden. Diese Daten werden in der Regel bereits durch eine dem Eye-Tracker beiliegende Software unter Nutzung von zeitlichen und räumlichen Grenzwerten zu Fixationen zusammengefasst. In dieser Arbeit wird davon ausgegangen, dass dieser Schritt bereits vollzogen wurde. Als Eingabe stehen also Fixationen inklusive Informationen über Position, Zeitpunkt, Dauer und Proband zur Verfügung. Außerdem muss für jede AOI bekannt sein, welchen Teil des Stimulus sie abdeckt, beispielsweise indem die Eckpunkte einer rechteckigen AOI gespeichert wurden.

Im nächsten Schritt werden mit Hilfe dieser Informationen für einen Stimulus alle Fixationen eines Probanden der AOI zugeordnet, in deren Bereich sie auftraten (siehe Abschnitt 5.2.1). An dieser Stelle kann direkt eine zeitliche Quantisierung mittels Temporal Binning stattfinden. Anschließend können AOI-Aufenthalte, die bisher durch einen Teilstring identischer Zeichen repräsentiert wurden, zu einem einzelnen Zeichen zusammengefasst oder je nach Länge herausgefiltert werden.

Danach findet der eigentliche Vergleich der Scanpaths mit einer der Metriken statt. Die Ergebnisse dieses Vergleichs ermöglichen eine Gruppierung der Scanpaths anhand ihrer Ähnlichkeit. Schließlich können die Ähnlichkeitswerte und die erzeugten Gruppen visualisiert werden, um das Verständnis der Ergebnisse zu erleichtern.

5.2 Scanpaths

Scanpaths stellen einen abstrakten Datentyp dar. Die folgenden Unterabschnitte beschäftigen sich mit der Repräsentation, Erzeugung, Vorverarbeitung und Kompression von Scanpathdaten.

5.2.1 Repräsentation und Erzeugung von Scanpaths

Bei den in Kapitel 4 vorgestellten verwandten Arbeiten wurden in der Regel Strings verwendet, in denen Zeichen wie etwa Buchstaben für Fixationen oder AOI-Aufenthalte stehen. Vorteile dieser Repräsentation sind die vereinfachte Darstellung im Vergleich zu Sakkaden-Vektoren und der gespeicherte Bezug zu den AOIs, in denen wiederum Informationen über deren Ähnlichkeit und Semantik enthalten sein können. Weiterhin können Strings direkt mit den verschiedenen String-Vergleichs-Algorithmen verarbeitet werden, weshalb dieses Prinzip mit einigen Erweiterungen übernommen wird. Dabei muss jedoch der Nachteil einer Quantisierung von räumlichen und zeitlichen Daten in Kauf genommen werden, die etwa bei Vektoren entfällt.

Die Scanpaths werden aus aufgenommenen Eye-Tracking-Daten gebildet, indem für jeden Probanden die zu diesem gehörenden Fixationen durchlaufen werden. Dabei wird chronologisch vorgegangen, in der Reihenfolge, in der die Fixationen in der Studie tatsächlich stattfanden. Für jede Fixation wird anhand ihrer Position bestimmt, in welcher AOI sie sich befand, und die Kennung dieser AOI in den Scanpath eingefügt. So referenzieren die Zeichen des Scanpath-Strings zunächst nur die AOI, in der die zu dem Zeichen gehörende Fixation stattgefunden hat.

5.2.2 Möglichkeiten zur Vorverarbeitung

Um die Ergebnisse eines Vergleichs unter Berücksichtigung weiterer Informationen oder besonderer Anforderungen zu verbessern oder die Berechnung zu beschleunigen, können die Scanpaths nach ihrer Erzeugung noch eine Reihe aus Vorverarbeitungsphasen durchlaufen. Diese werden im Folgenden vorgestellt und sind in Abb. 5.2 dargestellt.

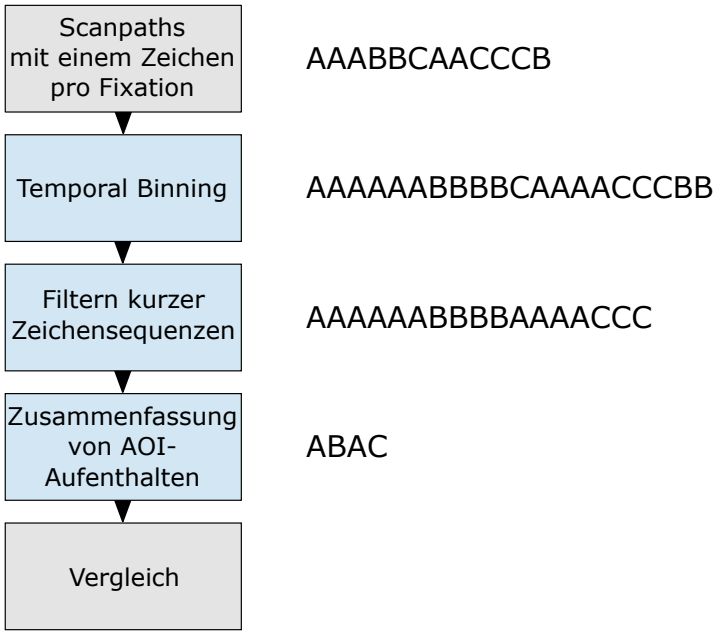


Abbildung 5.2: Vorverarbeitung der Scanpaths. Links sind die Schritte zu sehen, welche von den Pfaden durchlaufen werden. Jeder davon ist optional und es kann eine beliebige Kombination in der hier dargestellten Reihenfolge gebildet werden. Rechts ist ein Beispiel für einen Scanpath zu sehen, der alle Schritte durchläuft. Hier wirkte sich beim Temporal Binning die Dauer der Fixationen in den AOIs A und B so aus, dass sich die zu ihnen gehörenden Zeichen jeweils verdoppelt haben, während die Fixationsdauer in C nur für je ein Zeichen reicht. Im nächsten Schritt wurden Zeichenfolgen mit weniger als drei identischen Zeichen herausgefiltert. Zum Schluss wurden die verbleibenden AOI-Aufenthalte zu einem Zeichen zusammengefasst. Die Kombination von Binning und Filterung erlaubte eine Filterung der Daten nach Dauer statt Anzahl der Fixationen.

Wie bei ScanMatch [4] kann die Fixationsdauer mittels *Temporal Binning* einbezogen werden, durch eine der Zeitspanne entsprechende Wiederholung der AOI-Bezeichnung. Dazu wird eine bestimmte Zeit festgelegt, ab der ein zusätzliches Zeichen hinzugefügt werden soll. Dieser zeitliche Grenzwert beträgt standardmäßig 50 Millisekunden, kann aber auch je nach Samplingrate des Eye-Trackers oder Dauer der gemessenen Fixationen angepasst werden. Dann wird die Dauer der Fixation durch diesen Wert geteilt und das korrekt gerundete Ergebnis als Anzahl der einzufügenden Zeichen gewählt. Die Auswirkungen dieser zeitlichen Quantisierung wird in Abb. 5.2 gezeigt.

Falls vom Analysten für eine bestimmte Studie als sinnvoll erachtet, können kurze und dadurch weniger relevante AOI-Aufenthalte herausgefiltert werden. Als kurz gilt dabei ein Aufenthalt, der im String durch weniger als eine von Analysten festgelegte Anzahl von Zeichen repräsentiert wird. Für diesen Wert empfiehlt sich die typischerweise zu erwartende Zahl von Fixationen, bei der ein Aufenthalt noch keine Bedeutung für die Auswertung der Studie hat. Wurde Temporal Binning verwendet, kann eine Zeit in Millisekunden dividiert durch den zeitlichen Grenzwert des Binnings als Wert gewählt werden. Bei einem Grenzwert von drei Zeichen, was bei einem 50 Millisekunden Grenzwert beim Temporal Binning etwa einer Zeit von 150 Millisekunden entspricht, wird beispielsweise der String *AAAAAABBBB**C**AAAACCC**BB*** zu *AAAAAABBBBAAAACCC*. Durch diese Option kann Rauschen in den Daten reduziert werden, das etwa dadurch entstehen kann, dass manche der gemessenen Fixationen am Rand einer AOI außerhalb von dieser liegen, obwohl hauptsächlich das Objekt innerhalb der AOI betrachtet wurde. Abb. 5.3 zeigt, wie ein solcher Scanpath aussehen kann.

Durch die Wahl von höheren Grenzwerten kann die Analyse außerdem auf AOI-Aufenthalte beschränkt werden, die eine bestimmte minimale Aufmerksamkeit widerspiegeln. So werden Aufenthalte mit nur wenigen Fixationen oder, falls Temporal Binning genutzt wurde, von kurzer Dauer herausgefiltert. Diese können entstehen, wenn der Betrachter zwischen zwei entfernten AOIs wechselt, und Fixationen in dazwischen liegenden AOIs aufgezeichnet werden, obwohl der Betrachter mit dem dortigen Bildausschnitt nicht gedanklich beschäftigt war.

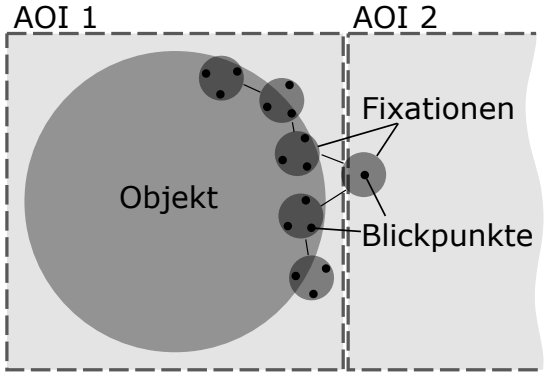


Abbildung 5.3: Ein Beispiel für die Auswirkung von Rauschen auf einen Scanpath. Während die meisten Blickpunkte innerhalb der AOI 1 liegen, die das betrachtete Objekt umgibt, wurde ein Blickpunkt im Bereich von AOI 2 gemessen. Er wurde als eigene Fixation angesehen und sorgt nun für eine Unterbrechung des ansonsten längeren Aufenthalts in AOI 1. Solche Abweichungen können durch das Filtern kurzer Sequenzen gleicher Zeichen vermindert werden.

Zuletzt können die erzeugten Strings vereinfacht werden, indem aufeinanderfolgend wiederholte Zeichen zu einem einzelnen zusammengefasst werden, wodurch beispielsweise aus *AAAAAABBB-BAAAACCC* der einfachere String *ABAC* entsteht. Dies kann sinnvoll sein, wenn bei der Analyse nur die Reihenfolge der betrachteten AOIs von Bedeutung ist, während die Aufenthaltsdauer nebensächlich für das Ergebnis der Studie ist. Bei dieser Option werden die Pfade oft stark verkürzt, was sich positiv auf den Aufwand und damit die Dauer der weiteren Verarbeitung auswirkt.

Alle diese Vorverarbeitungs-Operationen sind optional und werden nacheinander in der Reihenfolge durchgeführt, in der sie in diesem Abschnitt behandelt und in Abb. 5.2 gezeigt werden. Sie erlauben interessante Kombinationen, wie beispielsweise die von Temporal Binning mit dem Filtern kurzer Aufenthalte. Dadurch wird statt der Anzahl der Fixationen die Dauer der AOI-Aufenthalte für ein Passieren des Filters ausschlaggebend, so dass Aufenthalte unter einer gewissen Dauer herausgefiltert werden.

5.2.3 Scanpath-Kompression

Vor der Verarbeitung eines String-Paares durch einen der Vergleichsalgorithmen können die zu vergleichenden Pfade durch Kompression vereinfacht werden. Bei einer Kompression mehrerer Scanpath-Strings mit dem selben Wörterbuch werden in jedem Pfad identische Abschnitte, beziehungsweise Muster, durch den gleichen Wörterbuchindex ersetzt. Aus den beiden Scanpaths *ABCAC* und *BBCAB* könnten beispielsweise die Indexsequenzen *1, 4, 3* und *2, 4, 2* werden, wenn die Teilsequenz *BCA* im Wörterbuch bei Index 4 steht.

Bei der Kompression geht jedoch eine Zuordnung der Zeichen zu den AOIs verloren, weshalb bei Nutzung von Kompression eine Einbeziehung von AOI-bezogenen Informationen unmöglich wird. Beim Vergleich der komprimierten Strings durch beispielsweise die Levenshtein-Distanz können daher nur Kosten verwendet werden, die entweder konstant sind, oder anhand einer Funktion bestimmt werden (siehe Abschnitt 5.4.4). Die Kompression kann paarweise oder für alle Scanpaths mit dem selben Wörterbuch durchgeführt werden. Da es sich bei allen verglichenen Pfaden um den selben Stimulus handelt und daher auch die Teilstrategien der Probanden ähnlich sein werden, ist zweitere Möglichkeit vorzuziehen. Um sicherzustellen, dass bei jeder Durchführung der Kompression das Wörterbuch den selben Inhalt aufweist, können zunächst alle Pfade einmal komprimiert werden. Wenn die originalen Pfade danach ein zweites Mal komprimiert werden, sind bereits alle nötigen Wörterbucheinträge vorhanden und damit für alle Kompressionsvorgänge identisch. Die Reihenfolge der Kompressionen beim Aufbau des Wörterbuches beeinflusst dessen Inhalt. Daher sollte bei der Verwendung von Kompression in verschiedenen Vergleichen darauf geachtet werden, dass die Scanpaths immer gleich angeordnet sind.

5.3 Beurteilung der Scanpath-Vergleichs-Metriken

In Abschnitt 2.3 wurden verschiedene Methoden beschrieben, die zum Vergleich von Scanpaths verwendet werden können. Da es den Attention-Maps, Markov-Modellen sowie den Fixations-basierten Ansätzen an der Betrachtung der zeitlichen und vor allem sequenziellen Informationen mangelt, werden diese nicht weiter in Betracht gezogen. Sie sind nicht in der Lage, Strategien der Probanden anhand von Unterschieden in der Reihenfolge der Aktionen zu vergleichen. Vektor-basierte Methoden sind in dieser Hinsicht überlegen und erzielen aufgrund des Verzichts auf AOIs eine hohe Genauigkeit, da die Fixationspositionen direkt verwendet und nicht auf AOIs quantisiert werden. Sie können jedoch keine semantischen Informationen berücksichtigen. Die vorgestellten String-basierten Metriken verarbeiten Scanpaths, in denen sequenzielle Informationen über die AOI-Aufenthalte und teilweise auch zeitliche Informationen enthalten sind. Die verwendeten AOIs lassen sich zudem semantisch annotieren und können in Beziehungen zueinander gestellt werden. Daher wurden in dieser Arbeit nur diese Metriken auf ihre jeweiligen Vor- und Nachteile hin untersucht und verglichen. Eine Zusammenfassung der Ergebnisse findet sich in Tabelle 5.1.

Die Hamming-Distanz scheidet aus, da sie nur für Strings mit identischer Länge ausgelegt ist und andere Metriken wie die Levenshtein-Distanz flexiblere Möglichkeiten für die Festlegung von Kosten oder Punkten für ähnliche Stellen erlauben. Letztere ist zudem durch die häufige Verwendung in verwandten Arbeiten zu einer Standardmethode geworden. Statt ihr kann die Damerau-Levenshtein-Distanz gewählt werden, da sie lediglich eine Erweiterung darstellt, und bei passender Konfiguration die Levenshtein-Distanz berechnen kann. Der Algorithmus von Needleman und Wunsch eignet sich trotz seiner Komplexität gut für den Vergleich von Scanpaths, da bei ihm die Möglichkeit besteht, Ähnlichkeitsbeziehungen zwischen AOIs direkt in der Ersetzungsmatrix zu berücksichtigen. Der Smith-Waterman-Algorithmus sucht, anders als die eben besprochenen Metriken, nach lokalen Übereinstimmungen. Daher eignet er sich zwar zum Finden ähnlicher Teilstrategien, aber nicht zum Vergleich der Ähnlichkeit der Strategien selbst in ihrer Gesamtheit. Die Longest Common Subsequence ist aufgrund fehlender Parameter leicht zu verwenden und kann eine interessante Möglichkeit darstellen, falls ihre Leistungsfähigkeit nicht unverhältnismäßig schlechter ist als die der anderen Metriken.

5.4 Anpassungen und Erweiterungen der Metriken

Neben der bereits in Abschnitt 2.2.1 angesprochenen Problematik der peripheren Wahrnehmung und der dadurch geschwächten Aussagekraft der Blickpositionen, gibt es weitere Komplikationen beim Versuch, Scanpaths zu vergleichen [11]. So kann es bei allen Metriken zu Schwierigkeiten kommen, wenn sich die Länge der Pfade stark unterscheidet. Hier kann eine Normalisierung nach der maximalen Länge hilfreich sein. Unterbrechungen zwischen ansonsten langen ähnlichen Teilen der Scanpaths erschweren den Vergleich. Die Festlegung von Kosten und weiteren Parametern ist selten eindeutig festlegbar und beeinflusst das Ergebnis enorm, gewählte Parameter passen in der Regel nicht für alle Vergleiche in einer großen Menge an Scanpaths. Manche der im letzten Abschnitt verglichenen Metriken bieten Möglichkeiten für Verbesserungen hinsichtlich dieser Probleme bei der Anwendung

Metrik	Vorteile	Nachteile
Hamming-Distanz	- einfaches Prinzip - schnelle Berechnung in $\mathcal{O}(n)$	- Strings müssen die selbe Länge haben - weniger flexibel als andere Metriken - es werden AOIs benötigt
(Damerau-) Levenshtein-Distanz	- verschiedene Kosten für Einfügung, Ersetzung, Löschung und Vertauschung (Damerau-Levenshtein) wählbar, räumliche und semantische Ähnlichkeit der AOIs kann miteinbezogen werden - wird in vielen Arbeiten zum Vergleich von Scanpaths benutzt und stellt daher einen Quasi-Standard dar	- die Ersetzungskosten sind im Allgemeinen unabhängig von der Ähnlichkeit zweier AOIs - zeitaufwändig ($\mathcal{O}(m \cdot n)$) - es werden AOIs benötigt
Needleman-Wunsch	- sucht globale Ähnlichkeit - Ersetzungskosten sind abhängig von den ersetzten AOIs wählbar - ist der Standardimplementierung der Levenshtein-Distanz überlegen	- zeitaufwändig ($\mathcal{O}(max(m,n)^3)$) - es werden AOIs benötigt
Smith-Waterman	- kann Ähnlichkeiten in Teilstrategien finden	- sucht lokale Ähnlichkeit, nicht globale - zeitaufwändig ($\mathcal{O}(m \cdot n)$) - es werden AOIs benötigt
Longest Common Subsequence	- leicht zu implementieren und zu verwenden - keine Parameter notwendig	- beschränkte Alignment-Operationen im Vergleich zu Levenshtein - räumliche und semantische Ähnlichkeit der AOIs werden nicht betrachtet - zeitaufwändig ($\mathcal{O}(m \cdot n)$) - es werden AOIs benötigt

Tabelle 5.1: Vergleich der vorgestellten String-basierten Scanpath-Vergleichsmetriken anhand ihrer jeweiligen Vor- und Nachteile. Die Bezeichner *m* und *n* bei den angegebenen Laufzeiten stehen für die Längen der verglichenen Strings.

zum Vergleich von Scanpaths. Diese Möglichkeiten werden in den folgenden Unterabschnitten aufgeführt und erklärt.

5.4.1 Einbeziehung der räumlichen AOI-Distanzen

Bei der Berechnung der Levenshtein-Distanz werden in der Regel konstante Kosten veranschlagt. Dadurch werden bei der Ersetzungsoperation alle AOIs implizit als paarweise gleich ähnlich betrachtet. Stattdessen können die jeweiligen Ersetzungskosten abhängig von den beiden betroffenen AOIs gewählt werden.

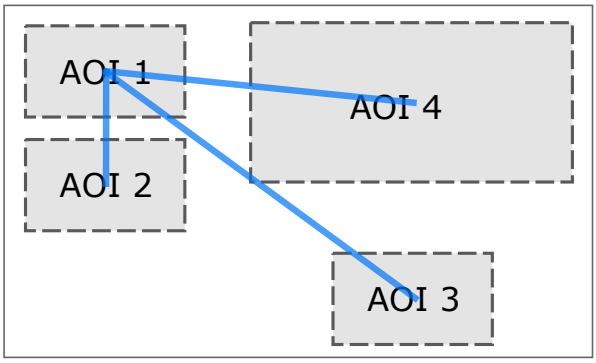


Abbildung 5.4: Räumliche Distanz von AOIs als Ähnlichkeitsmaß. Hier sind der Übersicht halber nur die Abstände von AOI 1 zu den anderen eingezeichnet (blau). Zur Berechnung der Abstände werden die AOI-Mittelpunkte verwendet. AOIs werden als umso ähnlicher aufgefasst, je näher sie beieinander sind. Damit wird berücksichtigt, dass trotz ähnlicher Suchstrategie der Probanden leichte Unterschiede in den betrachteten AOIs vorkommen können.

Eine Möglichkeit dafür sind die euklidischen Abstände der AOIs, die bei ScanMatch [4] für die Ersetzungsmatrix des Needleman-Wunsch Algorithmus verwendet wurden. Die Kosten für Vertauschungen bei Damerau-Levenshtein könnten auf die selbe Weise festgelegt werden, indem der Abstand der beiden in der Position vertauschten AOIs in den Wert einfließt. Das könnte die negative Auswirkung kleiner Unterschiede zwischen den Scanpaths auf das Ergebnis des Vergleichs verringern. Dabei kann je nach Wahl eines Faktors bestimmt werden, inwiefern solche Abweichungen toleriert werden sollen.

AOIs können eine beliebige Form besitzen, so sind auch dreidimensionale und sich dynamisch verändernde AOIs möglich. Dadurch kann die Berechnung der Distanz problematisch werden. Es bietet sich an, den Abstand der beiden AOI-Schwerpunkte zu nutzen, auch wenn diese Punkte beispielsweise bei nicht konvexen Polygonen außerhalb der AOI selbst liegen können. Abb. 5.4 veranschaulicht diese Abstände. Um Tendenzen im Blickverhalten eines Probanden, wie etwa der stärkeren Fokussierung auf die Mitte eines Bildes als auf den Rand, ausgleichen zu können, ist eine Normalisierung durch die Position der AOI innerhalb des Stimulus denkbar. Eine weitere Normalisierung kann auch Unterschiede in der Größe der AOIs und der damit verbundenen höheren Wahrscheinlichkeit für Fixationen in größeren AOIs berücksichtigen.

5.4.2 Einbeziehung semantischer Beziehungen zwischen den AOIs

Sofern es sich um semantische AOIs (siehe Abschnitt 2.2.5) handelt, können Informationen über die Beziehungen der AOIs untereinander für die Bestimmung von deren Ähnlichkeit genutzt werden. Ein Beispiel dafür ist ein Etikett auf einem Produkt im Supermarkt, wie Abb. 5.5 schematisch zeigt. Es kann durch eine eigene AOI innerhalb der Produkt-AOI repräsentiert werden, wobei eine *vertikale* Eltern-Kind-Beziehung zwischen beiden besteht. Mehre Produkte können in eine gemeinsame Gruppe eingeordnet werden, was eine *horizontale* Beziehung ergibt. Eine solche Beziehung bestünde auch

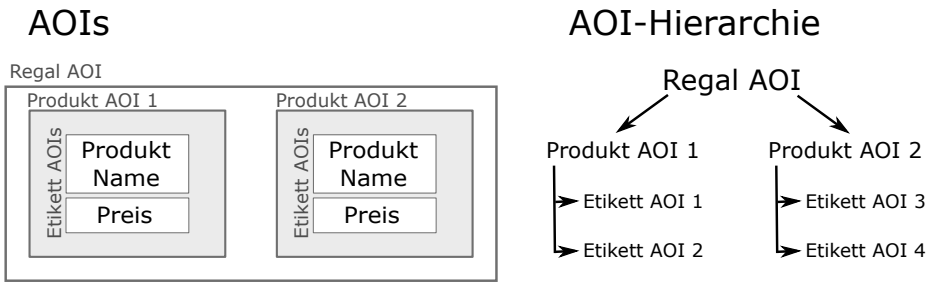


Abbildung 5.5: Ein einfaches Beispiel für semantische Beziehungen zwischen mehreren AOIs. Auf der linken Seite ist die Anordnung der AOIs zu sehen. Der Stimulus ist in diesem Fall ein Supermarktregal, in dem sich Produkte befinden. Auf diesen sind wiederum Etiketten angebracht, die den Namen und Preis der Produkte angeben. Für alle diese Objekte wurden passende Bereiche als ineinander verschachtelte AOIs festgelegt. Rechts ist die aus dieser Verschachtlung entstehende Hierarchie abgebildet.

zwischen zwei Etiketten auf dem selben Produkt, da die beiden entsprechenden Etiketten-AOIs zur selben Eltern-AOI gehören.

Je nach Ziel der Studie kann es sinnvoll sein, AOIs mit ähnlicher oder untergeordneter Semantik als *austauschbarer* zu betrachten als solche mit unterschiedlicher Bedeutung. Dementsprechend können beispielsweise die Ersetzungskosten der verschiedenen Stringvergleichs-Algorithmen angepasst werden. So kann berücksichtigt werden, dass etwa das Fixieren eines beliebigen Produktes ausschlaggebend und die Unterscheidung der einzelnen Produkte weniger relevant für die Analyse ist. Bei einem Übergang zwischen zwei Produkt-AOIs können auf diese Weise niedrigere Kosten angerechnet werden, wodurch die Scanpaths als ähnlicher gelten.

5.4.3 Datengetriebene Ähnlichkeitswerte für AOIs

Um einen Ähnlichkeitswert für zwei AOIs zu bestimmen, können die bei einer Studie aufgezeichneten Daten selbst verwendet werden. Solche datengetriebene Ansätze werden bisher unter anderem dafür genutzt, AOIs automatisch festzulegen [21]. Dabei werden Fixationen durch Clustering gruppiert und für die resultierenden Gruppen AOIs angelegt.

Für die Ähnlichkeit von AOIs entsprechend dem Blickverhalten von Probanden ist hingegen vor allem die Übergangshäufigkeit, beziehungsweise die Wahrscheinlichkeit eines Übergangs zwischen zwei AOIs von Bedeutung. Wie bei einem Markov-Modell nullter Ordnung (siehe Abschnitt 2.3.2) kann dabei betrachtet werden, wie wahrscheinlich es ist, dass eine AOI besucht wird. Soll dies noch in Abhängigkeit von der gerade betrachteten AOI geschehen, läge ein Markov-Modell erster Ordnung vor, bei dem die Übergangswahrscheinlichkeit analysiert wird. Ein Beispiel für die Übergangswahrscheinlichkeiten zwischen den AOIs wird in Tabelle 5.2 gezeigt. Die einzelnen Wahrscheinlichkeiten werden aus der Verteilung der Betrachtungszeit, der Fixationen oder der Übergänge hergeleitet. Eine Normalisierung kann auch abhängig von der Größe und Position der AOIs vorgenommen werden, um Tendenzen bei der Betrachtung auszugleichen.

AOI	A	B	C	D
A	0,3	0,3	0,2	0,2
B	0,3	0,1	0,4	0,2
C	0,2	0,4	0,3	0,1
D	0,2	0,2	0,1	0,5

Tabelle 5.2: Ein Markov-Modell erster Ordnung kann als AOI-Ähnlichkeitsmaß verwendet werden. Die Werte in der Tabelle geben die einzelnen Übergangswahrscheinlichkeiten zwischen zwei AOIs an. Sie wurden aus den tatsächlich aufgezeichneten Übergängen hergeleitet.

5.4.4 Variation von Metrik-Parametern über die Zeit

In manchen Fällen sind bei der Analyse der aufgezeichneten Daten bestimmte Zeitabschnitte besonders relevant oder vernachlässigbar. So beginnen die Probanden bei bildschirmbasierten Studien in der Regel mit einem Blick auf die Mitte des dargestellten Stimulus. Andererseits kann eine bestimmte Teilszene wichtiger für den Vergleich mehrerer Probanden sein, als das restliche Geschehen. Zudem werden die erkennbaren Strategien nach anfänglicher Ähnlichkeit oft im Verlauf der Aufnahme immer unähnlicher, auch aufgrund des sich akkumulierenden Rauschens in den Daten.

Beim Vergleich von Scanpaths können diese und weitere Fälle berücksichtigt werden, wenn die Kosten über den zeitlichen Verlauf der Daten mittels gewählter Funktionen gewichtet werden. Ein hoher beziehungsweise niedriger Funktionswert kann dafür sorgen, dass Unterschiede zwischen den Scanpaths in einem speziellen Abschnitt große oder vernachlässigbare Auswirkungen auf die insgesamt berechnete Ähnlichkeit haben. Dabei stehen verschiedene Möglichkeiten für die Betrachtung der Zeit zur Verfügung. Zunächst kann die Funktion als Parameter den absoluten oder den relativen Zeitpunkt einer Fixation betrachten. Bei Scanpaths mit unterschiedlicher Länge würde die Zeit bei einer relativen Betrachtung beispielsweise auf einen Wert zwischen Null und Eins normalisiert. Dadurch entspricht ein Wert von $0,5$ bei beiden Pfaden unabhängig von ihrer Länge jeweils der Mitte des Zeitverlaufs.

Falls Strings für die Repräsentation von Scanpaths verwendet werden, fehlt oft eine genaue Zeitangabe für Beginn und Ende der Fixationen. Diese Zeit kann nur anhand der Anzahl der Zeichen zwischen dem Anfang des Strings und einer bestimmten Stelle abgeschätzt werden. Dabei kann ein Zeichen je nach erfolgter Vorverarbeitung (siehe Abschnitt 5.2.1) für eine Fixation, einen AOI-Aufenthalt oder einen Zeitabschnitt einer Fixation stehen. Weiterhin können manche der bei der Studie aufgezeichneten Fixationen im gespeicherten Scanpath fehlen, da sie außerhalb aller festgelegten AOIs stattfanden oder herausgefiltert wurden.

Für die Gewichtungsfunktion selbst kommen verschiedene Alternativen in Frage (siehe Abb. 5.6). Möglich wäre etwa eine *konstante Funktion*, mit der die Parameter der Metriken zu jedem Zeitpunkt gleich gewichtet werden. Auf diese Weise kann auf eine Priorisierung verschiedener Zeitabschnitte verzichtet werden. Um beispielsweise zu Beginn aufgenommene Daten stärker zu gewichten, kann ein *Polynom*, eine *Logarithmus-* oder *Exponentialfunktion* genutzt werden. Diese können beliebig skaliert und kombiniert werden. Eine *abschnittsweise definierte Funktion* erlaubt eine unterschiedliche Gewichtungsfunktion innerhalb verschiedener Abschnitte.

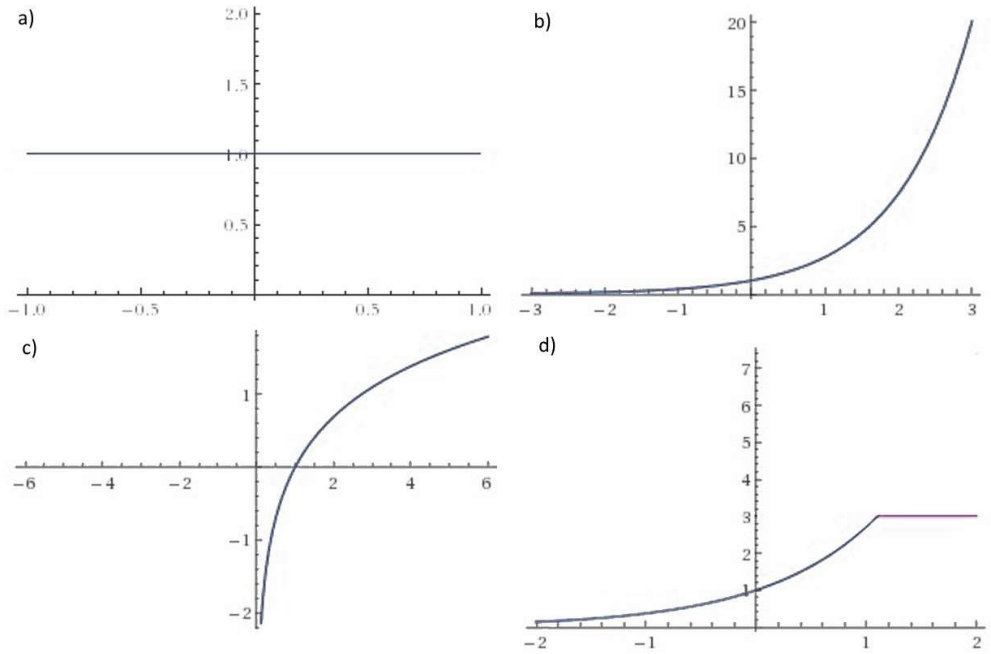


Abbildung 5.6: Möglichkeiten zur Festlegung der Gewichtungsfunktion. Eine konstante Funktion gewichtet jeden Abschnitt gleich (a). Exponentialfunktionen steigen schnell an (b), wohingegen logarithmische Funktionen immer weiter abflachen, ohne gegen einen Grenzwert zu konvergieren (c). Partiell definierte Funktionen erlauben beliebige Kombinationen. In Bild (d) wurde eine exponentielle Funktion mit einer konstanten so verbunden, dass sich eine stetige Funktion ergibt, Stetigkeit ist jedoch keine Voraussetzung. Die Bilder wurden erzeugt mit Wolfram Alpha[1].

Durch die veränderten Kosten und Punktzahlen ändert sich eventuell der minimale und maximale Wert, der durch den Algorithmus für zwei Scanpaths berechnet werden könnte, weshalb die Normalisierung der Ergebnisse angepasst werden muss (siehe Abschnitt 5.4.5).

5.4.5 Normalisierung der Ergebnisse

Sollen mehrere Paare von Scanpaths anhand ihrer jeweiligen Ähnlichkeit gruppiert werden, ist ein Wert für die Ähnlichkeit zweier Strings notwendig, der unabhängig von deren Länge ist. Die wird durch eine Normalisierung erreicht, welche im Allgemeinen folgendermaßen durchgeführt wird:

$$\frac{\text{<Resultat>} - \text{<minimal mögliches Resultat>}}{\text{<maximal mögliches Resultat>}} \tag{5.1}$$

Bei der Levenshtein-Distanz wird in vielen vorhandenen Arbeiten eine Normalisierung mit der Länge des längeren Strings vorgenommen [15]. Diese Methode ist jedoch nicht ausreichend, wenn aufgrund

[1] www.wolframalpha.com

der festgelegten Kosten Ergebniswerte entstehen, die größer als die Stringlänge sind, da sich dann für das normalisierte Ergebnis ein Wert größer als Eins ergibt. Stattdessen kann der maximale in der Berechnungstabelle vorkommende Wert zur Normalisierung herangezogen werden. Der auf diese Weise gewählte Wert steht für das schlechtest mögliche Ergebnis, das für die Ähnlichkeit der beiden Strings herauskommen kann.

Für den Algorithmus von Needleman und Wunsch bietet sich eine Normalisierung an, bei der das berechnete Ergebnis durch das Produkt aus dem Maximum der Ersetzungsmatrix und der Länge des längeren Strings geteilt wird. Diese Vorgehensweise wurde bei ScanMatch (siehe Abschnitt 4.1) angewandt und sorgt dafür, dass sich beim Vergleich zweier identischer Strings immer ein Wert von Eins ergibt.

Die Longest Common Subsequence zweier Strings kann auf keinen Fall länger sein als der kürzere der beiden. Daher bietet sich dessen Länge zur Normalisierung an, falls die lokale Ähnlichkeit der Strings im Vordergrund steht. Da zwei Scanpaths mit verschiedener Länge jedoch als eher unterschiedlich gelten sollen, wird hier eine Normalisierung anhand der Länge des längeren Strings verwendet. Ein Problem tritt auf, falls die Punktzahlen, im Falle dieser Metrik ein Wert von Eins für jedes Zeichen der LCS, dermaßen gewichtet werden, dass das Ergebnis des Algorithmus höher ist als die Länge des längeren Scanpaths. Dann schlägt die Normalisierung fehl, da Werte größer Eins herauskommen können, wenn die LCS eine entsprechend große Länge hat. Auch hier kann wie bei der Levenshtein-Distanz der größte Wert aus der Berechnungstabelle zur Normalisierung verwendet werden.

5.5 Gruppierung der Scanpaths nach Ähnlichkeit

Zur Gruppierung der Scanpaths wird hierarchisches agglomeratives Clustering genutzt (siehe Abschnitt 2.4.1). Der Vorteil dieser Technik ist, dass die Anzahl der zu erzeugenden Gruppen nicht im Voraus festgelegt werden muss. Stattdessen kann auch nach der Durchführung des Clusterings die erzeugte Hierarchie jederzeit auf einer beliebigen Höhe abgeschnitten werden, wodurch die Möglichkeit besteht, auch im Nachhinein Gruppen zu erzeugen. Dies kann unter anderem anhand einer Visualisierung der Hierarchie interaktiv vom Benutzer gesteuert geschehen.

Alternativen wären ein Top-Down-Ansatz sowie flaches Clustering. Ersteres teilt eine zu Anfang bestehende Gruppe mit allen Elementen immer weiter auf, bis nur noch ein Element pro Gruppe verbleibt. Diese Aufteilung ist jedoch aufwändiger zu implementieren. Flaches Clustering, etwa mittels k-Means-Algorithmus ermittelt keine Hierarchie und hat daher den Nachteil, dass die Anzahl der resultierenden Gruppen nicht im Nachhinein festgelegt werden kann.

Beim gewählten hierarchischen Clustering werden immer die zwei aktuell ähnlichsten Cluster zu einem neuen vereinigt. Dabei kann, falls gewünscht, jederzeit gestoppt werden, sobald die Ähnlichkeit zwischen allen möglichen Paaren von Clustern einen Grenzwert unterschreitet oder eine minimale Anzahl von Gruppen erreicht wurde. Die Ähnlichkeit zwischen zwei Clustern wird dabei bestimmt, indem je ein bestimmter Scanpath aus beiden gewählt wird und diese Scanpaths durch eine der Metriken verglichen werden. Für die Auswahl dieser repräsentativen Scanpaths gibt es zwei unterschiedliche Fälle. Bei Clustern mit nur einem Scanpath wird dieser ausgewählt. Sind mehrere Pfade im Cluster enthalten, wird ein geeigneter Repräsentant gesucht, der hier als *Most Central Scanpath* bezeichnet

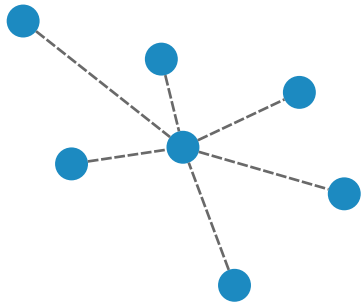


Abbildung 5.7: Veranschaulichung der Bestimmung des Most Central Scanpath. Es wird derjenige Scanpath als Repräsentant für den Cluster gewählt, bei dem die Summe der Abstände zu allen anderen Mitgliedern des Clusters minimal ist. Damit ist der Repräsentant am ähnlichsten zu allen Mitgliedern. In dieser Graphik wurden nur die Abstände des Most Central Scanpath zu den anderen Pfaden dargestellt.

wird (siehe auch Abb. 5.7). Dazu werden alle Scanpaths innerhalb des Clusters miteinander verglichen und derjenige ausgewählt, bei dem die Summe der Ähnlichkeiten zu allen anderen am höchsten ist. Diese Vorgehensweise ist sinnvoll, da keine unterschiedliche Priorität der Scanpaths vorliegt. In diesem Fall könnte der wichtigste Scanpath einer Gruppe als Repräsentant gewählt werden, wie es beim Clustering von Wörtern geschieht. Ein Vergleich der beiden zentralen Scanpaths zweier Cluster spiegelt zudem die Ähnlichkeit der Cluster genauer wider als der Vergleich der beiden ähnlichsten Elemente, der bei einer *Single-Linkage* vorgenommen würde.

5.6 Visualisierung der Ergebnisse

Da die bloßen Ergebnisse der Scanpath-Vergleiche auch nach einer Gruppierung nur schwer zu analysieren sind, kann eine geeignete visuelle Aufbereitung dem Benutzer das Verständnis der Daten deutlich erleichtern. In den folgenden Unterabschnitten wird eine Auswahl von Visualisierungen gezeigt, mit denen Ähnlichkeiten zwischen einer Menge von Elementen dargestellt werden können.

5.6.1 Vergleichstabelle

Die paarweisen Ähnlichkeiten der Scanpaths untereinander können durch eine Vergleichstabelle übersichtlich dargestellt werden. Ein Beispiel dafür zeigt Tabelle 5.3. Die Beschriftungen der Zeilen und Spalten enthalten dabei die Namen der Scanpaths, die beispielsweise den Pseudonymen der Probanden entsprechen können. Die Werte im Tabellenfeld $T[i, j]$ stehen dann für das Ergebnis des Vergleichs des i-ten Scanpaths mit dem j-ten Scanpath. Sofern die Metrik symmetrisch ist, also die Reihenfolge der beiden Eingabestrings keine Relevanz auf das Ergebnis des Vergleichs hat, ist auch die Vergleichstabelle symmetrisch. Diese Eigenschaft spart Zeit bei der Berechnung und Analyse. Eine zusätzliche farbliche Kodierung der Werte kann dem Benutzer einen besseren Überblick verschaffen.

	P1	P2	P3	P4	P5
P1	1	0,31	0,25	0,20	0,11
P2	0,31	1	0,47	0,28	0,17
P3	0,25	0,47	1	0,19	0,06
P4	0,20	0,28	0,19	1	0,38
P5	0,11	0,17	0,06	0,38	1

Tabelle 5.3: Vergleichstabelle der Ergebnisse eines Vergleichs von fünf Scanpaths mit einer gewählten Metrik. In diesem Beispiel wurden die Werte normalisiert, so dass ein Wert von 1 für zwei identische Pfade steht. Die Einfärbung der Zellen entspricht dem enthaltenen Wert, wodurch ein Vergleich der Werte einfacher wird. Die Symmetrie der Tabelle erfolgt aus der Symmetrie der verwendeten Metrik.

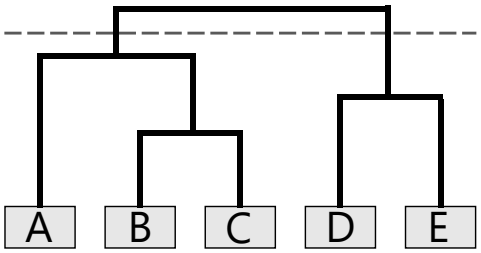


Abbildung 5.8: Diese Abbildung zeigt eine Hierarchie, die als Dendrogramm visualisiert wurde. Die gestrichelte graue Linie deutet einen Schnitt durch die Hierarchie an, mit dem diese in zwei kleinere Hierarchien geteilt wird. Die Elemente in diesen beiden Bäumen bilden zwei nach Ähnlichkeit von einander getrennte Gruppen.

5.6.2 Dendrogramm und Baumansicht

Als Visualisierung für die Gruppenhierarchie wird ein Dendrogramm verwendet (siehe Abschnitt 2.4.2). Es zeigt die Reihenfolge der beim Clustering vorgenommenen Gruppenvereinigungen und damit die relativen Ähnlichkeiten zwischen den Daten. Der Benutzer kann hier, wie im obigen Absatz angesprochen, eine Grenze festlegen, an der die Hierarchie in mehrere Gruppen aufgeteilt wird.

Ähnlich wie beim Dendrogramm wird bei der Baumansicht die Hierarchie der Scanpaths angezeigt. Allerdings können einzelne Cluster in der Ansicht interaktiv erweitert beziehungsweise zusammengeklappt werden, um nur die momentan für den Benutzer interessanten Daten zu zeigen. Details über die Gruppen können in einer extra Spalte angezeigt werden.

5.7 Evaluation der Leistungsfähigkeit der Metriken

Die zentrale Aufgabe dieser Arbeit ist der Vergleich verschiedener Metriken hinsichtlich ihrer Fähigkeit, die Ähnlichkeit von Scanpaths festzustellen. Dazu wird ein Konzept für die Evaluation der von den Metriken berechneten Ergebnisse benötigt. Diese Evaluation wird durch die vielen wählbaren Parameter erschwert, wie etwa die Optionen bei der Erzeugung der Scanpaths und die von vielen

Algorithmen verwendeten Kosten. Zudem ist es schwer, objektiv zu entscheiden, welche Scanpaths als ähnlicher gelten sollen als andere.

Um einen Vergleich der Metriken vornehmen zu können, werden daher Daten mit einer im Voraus abschätzbaren Ähnlichkeit verwendet. Diese Ähnlichkeit soll erkannt werden, indem nach dem Vergleich und einem Clustering der Ergebnisse Gruppen entstehen, die den erwarteten Ähnlichkeitsbeziehungen entsprechen. Die für die Beschaffung solcher Daten benutzten Ansätze und Möglichkeiten für die Auswertung der entstehenden Resultate werden in den folgenden Abschnitten erläutert.

5.7.1 Erzeugung von Testdaten

Als Benchmark für die vielen Möglichkeiten, Scanpaths zu vergleichen, soll eine Reihe von Testdaten dienen. Diese werden teils maschinell erzeugt und teils aus realen Aufzeichnungen gewonnen.

Ein praktischer Ansatz für die Erzeugung einer großen Menge von Testdaten ist die automatische Generierung von Scanpaths in mehreren Gruppen analog zur Vorgehensweise bei ScanMatch [4] und MultiMatch [7], siehe dazu auch die Abschnitte 4.1 und 4.2. Scanpaths können zufällig erzeugt und diese dann zufällig variiert werden, was zu Gruppen paarweiser Ähnlichkeiten führt. Die detaillierte Vorgehensweise hierfür wird in den folgenden Absätzen beschrieben.

Die Länge der zu erzeugenden Pfade wird zwischen zwei Grenzwerten zufällig mit Gleichverteilung gewählt, um verschieden lange Pfade zu erhalten. Wie die resultierenden Pfade aussehen, ist anhand zweier Beispielpfade und ihrer Varianten in Abb. 5.9 gezeigt. Zunächst werden dazu Punkte mit normalisierten Koordinaten genutzt. Diese haben Werte zwischen Null und Eins, die zufällig gleichverteilt gewählt und später auf die Ausmaße des Stimulus hoch skaliert werden. Mit zunehmender Länge wird die erwartete Differenz zwischen zwei zufällig generierten Pfaden immer größer, da es unwahrscheinlich ist, dass Fixationen beide Male in der selben Reihenfolge ähnliche Positionen einnehmen.

Um nun aus diesen unterschiedlichen Pfaden Gruppen mit untereinander ähnlichen Mitgliedern zu erhalten, werden Varianten der Pfade gebildet. Dazu werden die einzelnen Punkte um einen Wert verschoben, der durch eine normalverteilte Zufallsvariable bestimmt wird. Diese soll den Mittelwert Null haben, die Standardabweichung wird bei jedem neuen Scanpath stufenweise erhöht. Dadurch entstehen erwartungsgemäß Varianten, die immer kleiner werdende Ähnlichkeiten mit dem Originalpfad aufweisen. Eine hohe Ähnlichkeit zwischen Varianten, die aus unterschiedlichen Originalen gewonnen wurden, ist bei geringer Standardabweichung sehr unwahrscheinlich, da hierfür jede einzelne Fixation des einen Scanpaths nahe bei der entsprechenden Fixation des anderen liegen muss. Je höher die verwendete Standardabweichung ist, desto eher kann dieser Fall auftreten, wodurch es für Metriken schwerer wird, die Varianten korrekt dem Original zuzuordnen.

5.7.2 Aufzeichnung von Eye-Tracking-Daten mit bekannter Ähnlichkeit

Um die Leistungsfähigkeit mit realen Eye-Tracking-Daten zu testen, soll ein Versuch mit einer kleinen Gruppe von Probanden und einem Bildschirm-basierten Eye-Tracker durchgeführt werden. Dabei wird ähnlich wie bei MultiMatch vorgegangen, indem nacheinander Punkte auf dem Bildschirm angezeigt

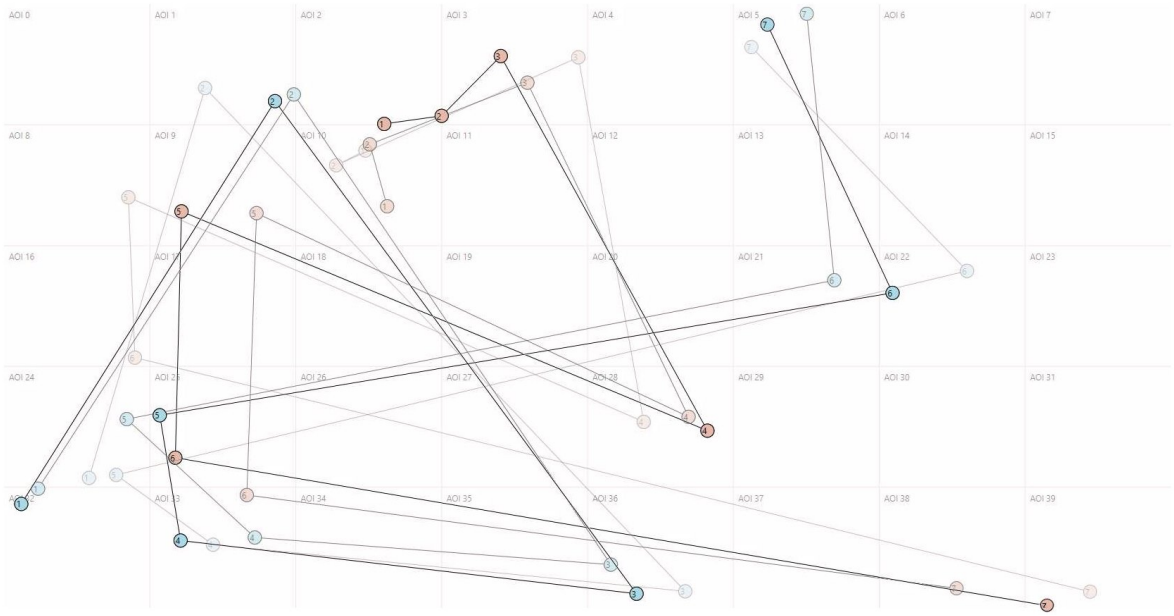


Abbildung 5.9: Zwei generierte Scanpaths (blau und orange) mit je zwei Varianten (semitransparent). Letztere wurden mit verschieden hoher Standardabweichung erzeugt, dargestellt durch eine höhere Opazität derjenigen Varianten, die eine geringere Abweichung zum Original haben. Nicht jede Abweichung ändert die getroffene AOI.

werden, die vom Probanden zu fixieren sind. Die Punkte geben einen von mehreren idealen Scanpaths vor, wodurch sich automatisch Gruppen mit zueinander ähnlichen Pfaden ergeben. Auf diese Weise sind die groben Ähnlichkeiten zwischen den Scanpaths bekannt. Diese Gruppen unterscheiden sich jedoch aufgrund des Einflusses des Probanden und der vorhandenen Messungenauigkeit des Eye-Trackers. Die Muster-Pfade werden dabei auf die selbe Weise erzeugt, wie zuvor die zufällig automatisch generierten Testdaten, allerdings ohne Varianten, da diese bei der Aufnahme durch die Testpersonen geschaffen werden.

5.7.3 Verwendung von Daten aus einer externen Studie

Als dritte Quelle für die zur Evaluation verwendeten Daten dient ein Datensatz aus einer externen Studie [20]. Diese wurde mit dem Ziel durchgeführt, einen Standard für die Evaluation von Eye-Tracking Visualisierungen und Analysewerkzeugen zu schaffen. Erhoben wurden die Aufzeichnungen der Probanden beim Betrachten verschiedener Videos. Die Daten eignen sich aufgrund der Tatsache, dass die Probanden in machen der Aufzeichnungen zwei verschiedene Aufgaben hatten. Daher sollten sich nach einem Clustering zwei etwa gleich große Gruppen von Scanpaths ergeben.

5.7.4 Auswertung der Ergebnisse

Durch die getroffenen Maßnahmen bei der Generierung und Aufzeichnung der Daten sind implizite Gruppen von Scanpaths vorhanden. Die Aufgabe der Vergleichsmetriken besteht nun darin, die Pfade korrekt zu diesen Gruppen zuzuordnen. Zu diesem Zweck werden die Scanpaths nach der Durchführung aller möglichen Vergleiche mit einer Metrik mittels hierarchischem agglomerativem Clustering gruppiert. Da die Anzahl der Gruppen bekannt ist, kann diese als Abbruchskriterum beim Clustering verwendet werden.

Die Erfolgsquote bei dieser Gruppierung dient dann als Indikator für die Leistungsfähigkeit der jeweiligen Metrik unter der festgelegten Konfiguration. Es werden also die korrekten Zuordnungen gezählt und deren Anteil an der Gesamtzahl zur Bewertung ausgegeben. Hierfür werden zunächst die jeweils als korrekt geltenden Scanpaths für jede Gruppe festgelegt. Dazu wird derjenige Original-Scanpath als Gruppeneigentümer gewählt, dessen Variationen am häufigsten in ihr vorkommen. Wurden auf diese Weise mehrere Gruppen vom selben Pfad beansprucht, wird für ihn die Gruppe mit den meisten seiner Variationen übernommen. Die restlichen Gruppen werden analog verteilt. Somit wird sichergestellt, dass jeder Scanpath nur zu einer Gruppe gehören kann. Nun können alle korrekten Zuordnungen gezählt und die Erfolgsquote berechnet werden. Durch diese Vorgehensweise können alle Scanpath-Optionen, die Metriken und deren Parameter in verschiedenen Konstellationen getestet werden, bis die beste Kombination gefunden wurde.

6 Implementierung

In diesem Kapitel wird auf die Umsetzung des erarbeiteten Konzepts in einer prototypischen Implementierung eingegangen. Abb. 6.1 zeigt das Hauptfenster der Anwendung. Die folgenden Abschnitte erläutern die Herkunft und Verwaltung der benötigten Daten, sowie deren Verarbeitung und die anschließende Visualisierung der Ergebnisse.

6.1 Verwendete Programme und Tools

Der Prototyp wurde in der Programmiersprache C# entwickelt, um die Kompatibilität zu bestehenden Komponenten zu erleichtern. Als Entwicklungsumgebung wurde das Microsoft Visual Studio 2013 benutzt. Die Datenbank wurde auf einem Microsoft SQL Server verwaltet.

6.2 Nutzung einer Datenbank für Studiendaten

Bei der Planung und Durchführung einer Studie fällt eine vielfältige Menge von Daten an. Dazu gehören beispielsweise Informationen über Studienteilnehmer, die verwendeten Stimuli und deren jeweilige AOIs. Nach der Aufnahme der Eye-Tracking Daten liegen diese als Rohdaten oder auch zusammengefasst als Fixationen vor. Um diese Daten effizient zu verwalten, bietet sich die Verwendung einer relationalen Datenbank an. Wie die Struktur einer solchen Datenbank aussehen kann, ist in Abb. 6.2 zu sehen. Die in dieser Implementierung verwendete Datenbank wurde von früheren Projekten ohne weitere Änderungen übernommen [14].

Dabei enthält jede Tabelle einen Typ von Daten, etwa Fixationen, und kann Beziehungen zu anderen Tabellen haben. So kann beispielsweise für jede Fixation die Zugehörigkeit zu einem Probanden und einem Stimulus in einem Tabellenfeld gespeichert werden. Durch die vorhandenen Informationen über die AOIs des Stimulus können die Fixationen auch diesen zugeordnet werden.

6.3 Verwaltung der Scanpath-Daten

Um den eigentlichen Pfad und über ihn hinausgehende Metainformationen strukturiert speichern zu können, wurde eine eigene Klasse implementiert. Sie enthält Angaben über Proband, Stimulus, AOIs und die getätigten Vorverarbeitungen (siehe Abschnitt 6.3.5). Damit sind die gespeicherten Scanpaths unabhängig von ihrer Herkunft verwendbar. In den folgenden Abschnitten werden die Herkunft und der Umgang mit den Scanpath-Daten erläutert.

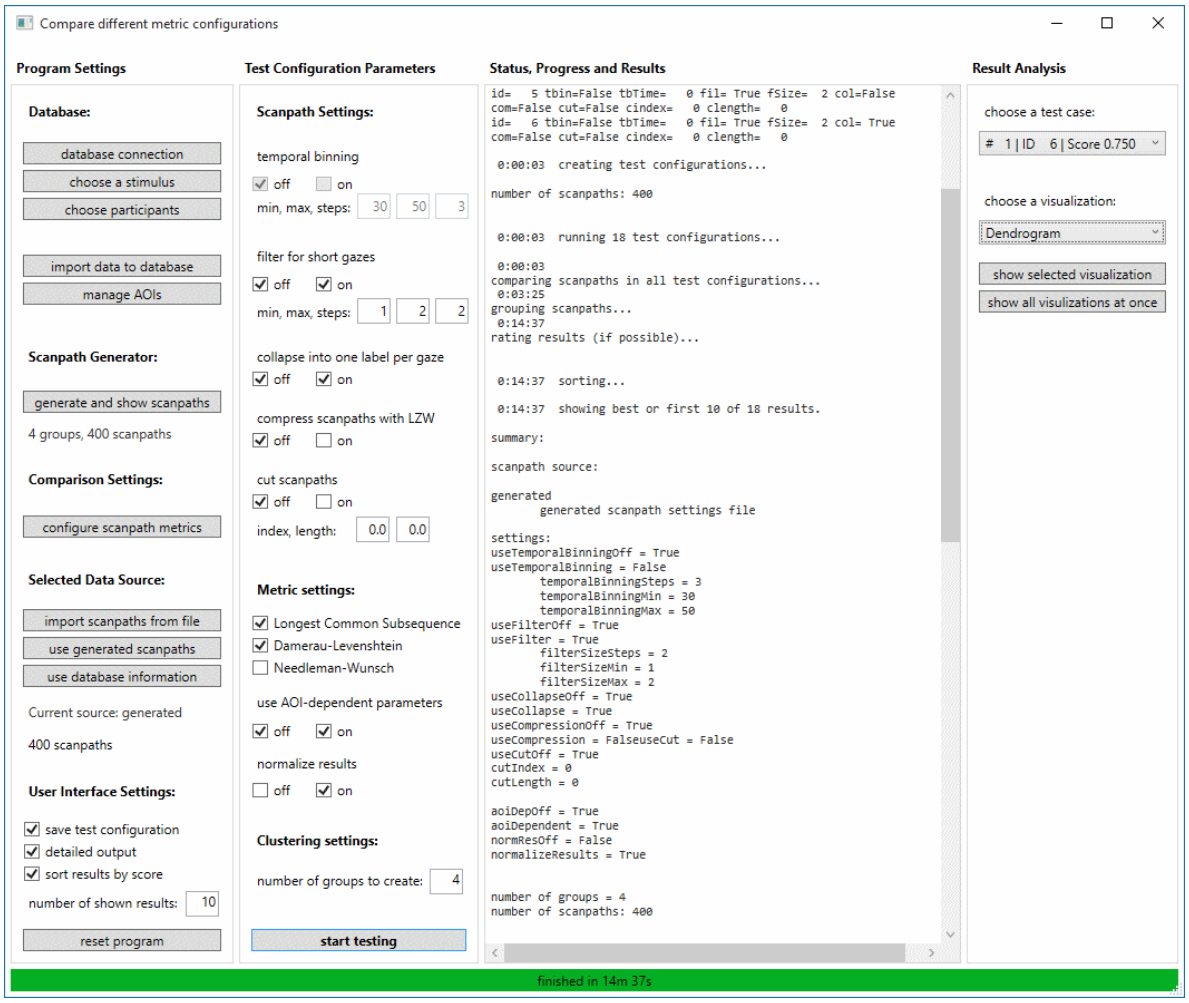


Abbildung 6.1: Das Hauptfenster des Prototypen. Hier kann unter anderem die Quelle der zu verarbeiteten Daten ausgewählt werden. In der zweiten Spalte ist eine Wahl der Parameterbereiche möglich, für welche eine entsprechende Anzahl an Testkonfigurationen erzeugt wird. Eine Konsole zeigt in Kombination mit der Fortschrittsleiste am unteren Rand den aktuellen Status der Verarbeitung an. Rechts kann nach deren vollständiger Durchführung für eine beliebige Testkonfiguration eine Visualisierung ausgewählt und anzeigt werden.

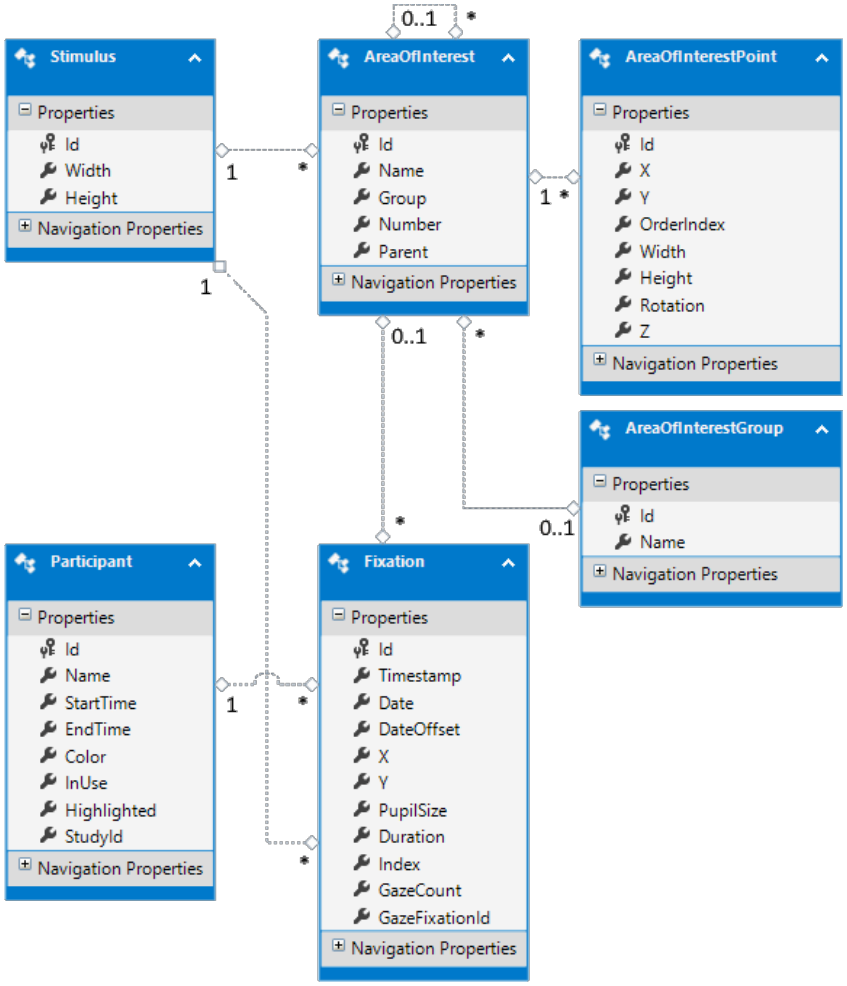


Abbildung 6.2: Die in dieser Arbeit relevanten Datenbanktabellen zur Verwaltung der Studiendaten, dargestellt als Entity-Relationship-Diagramm. Das Bild wurde erstellt mit Microsoft Visual Studio 2013 und zur Verbesserung der Lesbarkeit die Schrift nachbearbeitet.

6.3.1 Repräsentation

Wie bereits in Abschnitt 5.2.1 erwähnt, haben sich Strings zur Repräsentation von Scanpaths etabliert. Dadurch können sie direkt von den Stringvergleichs-Algorithmen verarbeitet werden. Um die Handhabung der Scanpaths in der Implementierung zu erleichtern, wurde in dieser Arbeit stattdessen eine Liste benutzt, die für jede Fixation die ID der zugehörigen AOI als Integer enthält. Dadurch ergeben sich praktische Vorteile, wie die Umgehung von Limitationen bei der Benutzung einer einzelnen Ziffer oder eines Buchstaben bei mehr als zehn beziehungsweise 26 verschiedenen AOIs. Eine Umwandlung von und zu herkömmlichen Zeichenketten ist dabei ohne weiteres möglich, wobei darauf geachtet werden muss, dass ein String, der nicht aus AOI-IDs besteht, eine Verwendung von Ähnlichkeitswerten der AOIs ausschließt. Diese könnten jedoch in einer späteren Erweiterung für jedes Paar an Wörterbucheinträgen berechnet oder manuell festgelegt werden.

6.3.2 Import aus einer Datei

Scanpaths können auch aus einer Datei importiert werden. In diesem Fall stehen allerdings keine weiteren Informationen über die Studie zur Verfügung, was sich auf die Möglichkeiten zum späteren Vergleich auswirkt. Die zu importierende Datei enthält Zahlen, die für die Fixationen eines Scanpaths stehen und jeweils die AOI angeben, in der die Fixation sich befand. Als Dateiformat werden zwei Alternativen unterstützt: Die einfache Variante besteht lediglich aus Zeilen mit je einem Scanpath, die in Form einer Folge von Ganzzahlen vorliegen müssen. Letztere müssen durch ein Leerzeichen oder Semikolon getrennt sein. Um Metainformationen des Scanpaths, wie beispielsweise den Namen des Probanden zu übergeben, der auch als Bezeichnung für den Scanpath dient, können Informationen in einer festgelegten Reihenfolge und durch Rautezeichen getrennt vor dem Beginn der Zahlenfolge stehen.

Da nur eine Reihe von AOI-Bezeichnern vorliegt, kann ein importierter Scanpath nicht durch Temporal Binning mit Zeitinformationen angereichert werden. Allerdings kann schon vor der Erstellung der Datei eine entsprechende Vorverarbeitung erfolgen. Aufgrund der nicht vorhandenen Informationen über die Ähnlichkeiten der AOIs müssen außerdem die Kosten und Punktzahlen bei den Metriken manuell festgelegt werden (siehe Abschnitt 6.4).

6.3.3 Generierung zufälliger Pfade

Wie in Abschnitt 5.7.1 für die Evaluation beschrieben, können die Pfade auch unter Nutzung zufälliger Punkte generiert werden. Weitere Anwendungsmöglichkeiten dieser Option sind beispielsweise Tests der Visualisierungen oder der Skalierung der Metriken für eine große Zahl oder Länge von Scanpaths. Die dafür verwendbare Benutzeroberfläche wird in Abb. 6.3 gezeigt. Wie die generierten Scanpaths aussehen können, zeigen die Abbildungen 5.9 und 7.2.

6.3.4 Erzeugung aus der Datenbank

Um Scanpaths aus den in der Datenbank gespeicherten Fixationen zu erzeugen, müssen diese zunächst der AOI zugeordnet werden, in der sie auftrat. Das geschieht anhand der für jede AOI in der Datenbank enthaltenen Informationen über die von ihr abgedeckte Fläche über dem Stimulus. In diesem Prototyp werden vorerst nur rechteckige AOIs unterstützt. Nach dieser Zuordnung werden ein Stimulus und eine Auswahl von Probanden festgelegt, anschließend wird für jeden Proband der zu ihm gehörende Scanpath erzeugt. Bereits in diesem Schritt erfolgt die Anwendung des optionalen Temporal Binning, mit einer vom Benutzer gewählten Zeitspanne. Die Dauer jeder Fixation wird dann durch diese Zeit geteilt, das mathematisch korrekt gerundete Ergebnis bestimmt dann die Anzahl der Zeichen, durch die diese Fixation im Scanpath repräsentiert wird.

6.3.5 Vorverarbeitung und Vereinfachung

Unabhängig von der Herkunft eines Scanpaths können die im Konzept erläuterten Vorverarbeitungsschritte *Filtern* und *Zusammenfassen* angewandt werden. Bei der Filterung kurzer Sequenzen

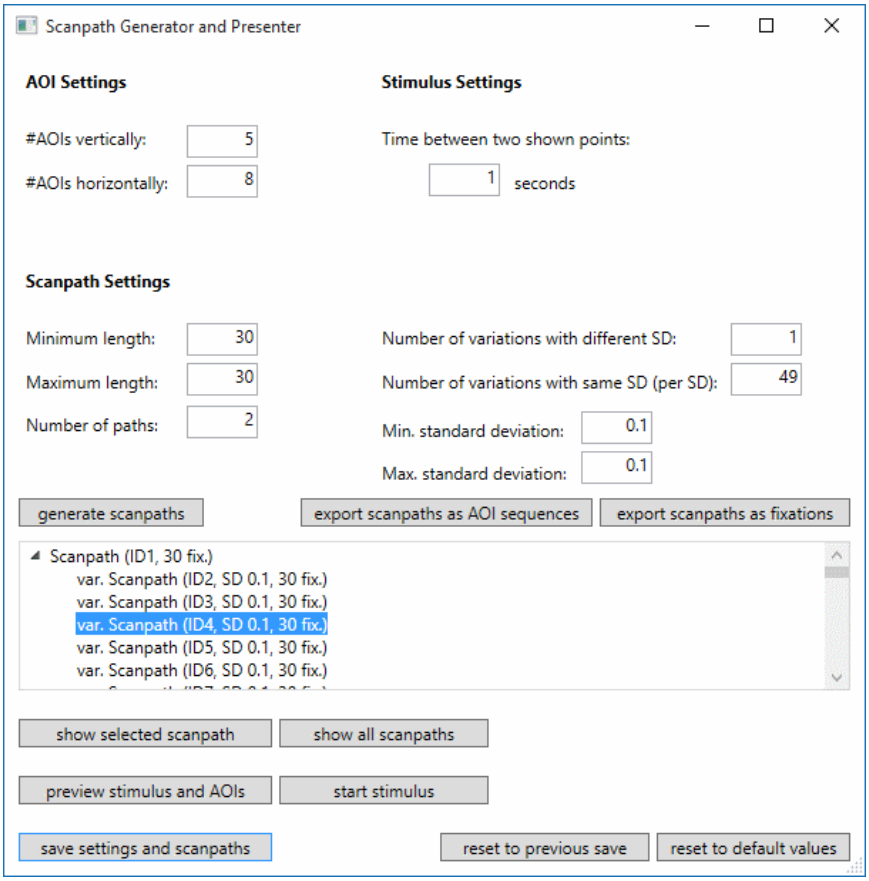


Abbildung 6.3: Das Konfigurationsfenster für die randomisierte Scanpath-Generierung. Diese Oberfläche ermöglicht die Angabe der gewünschten vertikalen und horizontalen Anzahl der gitterförmigen AOIs und die Wahl verschiedener Optionen für die Scanpath-Erzeugung. So können Varianten mit verschiedener Standardabweichung generiert werden, wobei auch für jeden Wert mehrere Varianten möglich sind. Buttons im unteren Bereich des Fensters führen zu verschiedenen Anzeigen des markierten oder aller Scanpaths.

identischer Zeichen werden jene mit einer Länge unter einem vom Benutzer gesetzten Grenzwert aus dem Pfad entfernt. Die Zusammenfassung ersetzt aufeinanderfolgende identische Zeichen durch ein einzelnes, um nur je ein Zeichen pro AOI-Aufenthalt im Pfad zu belassen.

6.3.6 Kompression

Für die Kompression der Scanpaths wurde der Algorithmus von Lempel, Ziv und Welch (LZW-Algorithmus) [45] gewählt, eine Verbesserung des LZ78 [51]. Er findet unter anderem beim verbreiteten Bildformat *Graphics Interchange Format (GIF)* Anwendung. Bei der Kompression einer Zeichenkette mit diesem Algorithmus wird solange auf ein *Wörterbuch* zugegriffen, bis der aktuell zu verarbeitende Abschnitt nicht mehr darin vorkommt. Dieser wird dann hinzugefügt und der Vorgang fortgesetzt.

Das Ergebnis ist schließlich eine Sequenz von Verweisen auf Wörterbucheinträge, durch die der ursprüngliche String verlustfrei wiederhergestellt werden kann.

Der Quellcode für die LZW-Kompression wurde zum Teil von Rosetta Code[1] übernommen und angepasst. Da im Gegensatz zum Original nicht Zeichenketten sondern Listen mit Zahlen komprimiert werden sollen, wurden als Schlüssel für das Wörterbuch entsprechend Zeichenketten aus den jeweiligen Zahlen und dazwischenliegenden Trennzeichen gebildet. Das Ergebnis der Kompression ist eine Liste von Ganzzahlen, die von den Metriken direkt wie ein Scanpath verwendet werden kann. Dabei entfällt jedoch die Möglichkeit, AOI-Informationen in den Scanpath-Vergleich einfließen zu lassen, da kein Bezug mehr zwischen den Zahlen im Pfad und den AOIs besteht. Auch manuell eingetragene AOI-abhängige Parameter lassen sich daher nicht nutzen (siehe Abschnitt 6.4). Die dazugehörigen Optionen werden vom Programm automatisch deaktiviert.

Die Kompression der Scanpaths findet im Anschluss an die restlichen Vorverarbeitungsschritte statt, da sonst andere Vorverarbeitungsschritte unmöglich würden. Temporal Binning kann nach einer Kompression nicht angewandt werden, da kein direkter Bezug zu den Fixationen besteht. Eine Filterung oder Zusammenfassung der Fixationen ist aus dem selben Grund ebenfalls nicht mehr möglich. Vor der eigentlichen Kompression der Scanpaths werden zunächst alle Pfade einmal komprimiert, um ein einheitliches Wörterbuch zu schaffen. Die Reihenfolge dieser Kompressionen beeinflusst die Einträge, daher sollte sie bei mehreren Vergleichsdurchläufen identisch sein. Nach dieser Vorbereitung werden die originalen Pfade erneut unter Benutzung dieses Wörterbuches komprimiert und die dabei entstehenden Pfade als Daten für den anstehenden Vergleich genutzt.

6.4 Module für den Vergleich von Scanpaths

Von den im Grundlagenkapitel vorgestellten Metriken wurden im Konzept drei ausgewählt, die sich als besonders geeignet für diese Arbeit herausstellten. Dies sind die Damerau-Levenshtein-Distanz, der Algorithmus von Needleman und Wunsch sowie die Longest Common Subsequence. In diesem Abschnitt werden die implementierten Anpassungen und Erweiterungen der Metriken für den Vergleich von Scanpaths behandelt.

6.4.1 Needleman-Wunsch

Der Algorithmus von Needleman und Wunsch wurde analog zur Vorgehensweise bei ScanMatch [4] implementiert (siehe auch Abschnitt 4.1). So werden als Werte für die Ersetzungsmatrix die euklidischen Abstände zwischen den AOIs verwendet, mit einem Grenzwert, ab dem die Werte negativ werden. Dieser beträgt die doppelte Standardabweichung der Längen aller Sakkaden, die zum aktuellen Stimulus vorhanden sind. Durch diese Maßnahme kann die Gap Penalty auf Null gesetzt werden. Da bei aus Dateien importierten oder randomisiert generierten Scanpaths keine Informationen über die Sakkaden der Probanden vorliegen, werden in diesen Fällen nur positive Werte und eine Gap

[1] http://rosettacode.org/wiki/LZW_compression#C.23

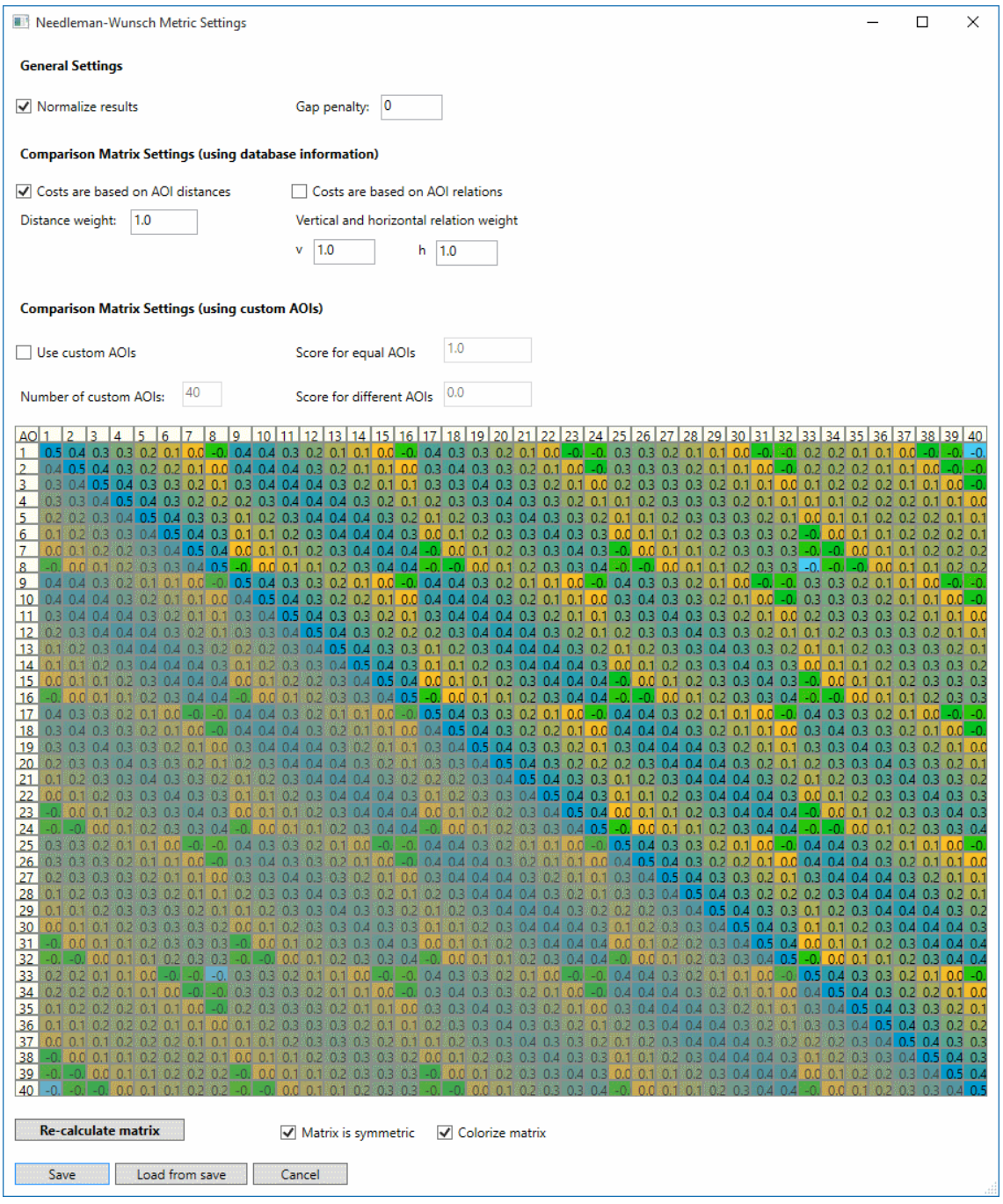


Abbildung 6.4: Einstellungen für den Needleman-Wunsch-Algorithmus. Die hier gezeigte Ersetzungsmatrix enthält Werte, die aus den Abständen der 8 mal 5 gitterförmigen AOIs berechnet wurden, die in der Evaluation Verwendung fanden. In diesem Fall wäre eine manuelle Eingabe mit einigem Aufwand verbunden, auch unter Ausnutzung der Symmetrie. Eine Einfärbung der Werte nach der Höhe der vergebenen Punktzahlen sorgt für einen besseren Überblick (siehe auch Abschnitt 6.7.1).

Penalty von -1 verwendet. Die Abstände werden zwischen den Mittelpunkten der AOIs berechnet, wobei rechteckige Bereiche und Polygone unterstützt werden.

Alternativ können alle Ähnlichkeitswerte auch manuell vom Benutzer eingetragen werden. Die Benutzeroberfläche dafür ist in Abb. 6.4 zu sehen. Textfelder bieten die Möglichkeit, die Punktzahlen abhängig von der Gleichheit zweier AOIs für die gesamte Tabelle festzulegen. Außerdem ist eine Bearbeitung jedes Tabellenfeldes möglich, bei aktivierter Symmetrie der Tabelle werden Änderungen automatisch gespiegelt. Eine Einfärbung der Tabellenfelder in Abhängigkeit von der Höhe des Wertes verbessert den Überblick. Dabei werden die Werte vom niedrigsten bis zum höchsten mit verschiedenen Farben hinterlegt, die je nach Größe der Werte aus einem Bereich zwischen zwei Farben gewählt werden. Es wurden zwei zueinander annähernd komplementäre Farben mit ähnlicher Luminanz gewählt, um eine feine Abstufung zu ermöglichen und die Lesbarkeit nicht zu beeinträchtigen.

Durch die manuelle Angabe der Ähnlichkeitswerte können Scanpaths unter Verwendung von AOI-abhängigen Parametern verglichen werden, auch wenn dem Programm keine vollständigen Informationen über die AOIs vorliegen. Dies ist etwa dann der Fall, wenn die Scanpaths aus einer Datei importiert wurden und die Datenbank keine Informationen über die verwendeten AOIs enthält. Außerdem können durch die manuelle Festlegung der Punktzahlen beliebige Ähnlichkeitskriterien für die AOIs beim Vergleich genutzt werden, beispielsweise abhängig von der Farbe der AOIs.

Je nach semantischer Beziehung der AOIs können die Werte noch durch einen wählbaren Faktor gewichtet werden. Es wird hier nach horizontaler und vertikaler Verwandtschaft unterschieden, für welche jeweils verschiedene Faktoren gewählt werden können. Siehe dazu auch Abschnitt 5.4.2.

Die Gap-Penalty kann ebenso frei gewählt werden. Bei Nutzung von AOI-abhängigen Ähnlichkeitswerten mit Grenzwert kann sie auf Null gesetzt werden [4]. Zuletzt kann eine Normalisierung der Ergebnisse vom Benutzer aktiviert oder deaktiviert werden.

6.4.2 Damerau-Levenshtein

Entsprechend der Implementierung des Algorithmus von Needleman und Wunsch wurde die Berechnung der Levenshtein-Distanz um AOI-abhängige Kosten erweitert. Diese können auch in diesem Fall anhand der Distanzen oder semantischen Beziehungen der AOIs untereinander berechnet werden. Bei distanzbasierten Kosten werden diese anhand der Stimulusdiagonalen normalisiert und dann mit den Kosten für Einfügen, beziehungsweise Löschen, multipliziert. Sie liegen damit im Bereich zwischen Null und dem Wert für die Einfüge- und Löschkosten. Die Einbeziehung der AOI-Informationen betrifft nur die Kosten für Ersetzung und die optional deaktivierbare Transposition.

Auch bei dieser Metrik kann die Kostentabelle manuell bearbeitet werden. Die hierfür zur Verfügung gestellte Benutzeroberfläche ähnelt der des Moduls für den Needleman-Wunsch-Algorithmus, die in Abb. 6.4 gezeigt ist. In diesem Fall ist die Kostentabelle um eine Reihe und Spalte erweitert. Die letzte Reihe und Spalte enthalten die Einfüge- beziehungsweise Löschkosten für jede AOI. Dem Benutzer stehen statt den Textfeldern für Punktzahlen drei Felder für die Kosten für Einfügen und Löschen, Ersetzen und die optionale Transposition zur Verfügung. Einfüge- und Löschkosten werden in jedem Fall als konstante Werte vom Benutzer gewählt, standardmäßig betragen beide den Wert Zwei.

6.4.3 Longest Common Subsequence

Für die Ermittlung der Ähnlichkeit von zwei Scanpaths durch die Longest Common Subsequence reicht die Berechnung der Länge dieser Sequenz aus, wofür der in Abschnitt 2.3.4 gezeigte Algorithmus implementiert wurde. Durch den Verzicht auf die Bestimmung der LCS selbst wird der Algorithmus einfacher und daher weniger zeitaufwändig in der Ausführung. Dabei kommen keine AOI-bezogenen Parameter wie bei den anderen beiden Metriken zum Einsatz, da lediglich die Länge der LCS bestimmt wird, was bedeutet, dass jedes Zeichen in dieser Sequenz einen Punkt zum Ergebnis beiträgt. Die für die Ähnlichkeit zweier Zeichen vergebenen Punkte können jedoch durch eine Funktion gewichtet werden, wodurch auch andere Werte als Eins zu einer *Pseudolänge* der LCS aufaddiert werden können (siehe Abschnitt 6.5). Es kann außerdem wie bei den anderen Metriken festgelegt werden, ob eine Normalisierung nach der Länge des größeren Scanpaths vorgenommen werden soll.

6.5 Variation von Metrik-Parametern über die Zeit

Da zeitliche Informationen in den Scanpaths höchstens indirekt durch Temporal Binning vorhanden sind, musste eine andere Möglichkeit für die Bestimmung der vergangenen Zeit an einer Stelle des Pfades gefunden werden. Als Annäherung für diese Zeit dient die Anzahl der Zeichen, die zwischen dem Beginn des Scanpaths und dieser Stelle stehen. Dieser Wert kann durch die Gesamtlänge normalisiert werden, indem die Anzahl der vorangegangenen Zeichen durch die Gesamtzahl, also die Länge des Scanpaths, dividiert wird.

Kosten oder Punktzahlen der Vergleichsmetriken werden dann gewichtet, indem bei jeder Berechnung eine Funktion in Abhängigkeit von der angenäherten Zeit berechnet und ihr Ergebnis als Skalierungsfaktor für den jeweils genutzten Metrik-Parameter verwendet wird. Es wurden dafür folgende Funktionen implementiert:

$f_1(t) = a \cdot t + b$

$f_2(t) = a \cdot log(t) + B$

$f_3(t) = a \cdot e^t + b$

$f_4(t) = a \cdot t^3 + b \cdot t^2 + c \cdot t + d$

$f_5(t) = 1$

Bei der Wahl der Parameter muss darauf geachtet werden, dass negative Werte zu unerwünschten Ergebnissen führen können. So kommt es dann beispielsweise zu einem umso geringeren Wert für die Levenshtein-Distanz, je unterschiedlicher die Scanpaths sind.

6.6 Clustering der Scanpaths

Nach dem Vergleich aller Scanpaths sollen diese zu Gruppen zusammengefasst werden, wozu Clustering angewandt wird. Implementiert wurde für diesen Zweck hierarchisches agglomeratives Clustering, wie in Abschnitt 5.5 beschrieben. Alle dabei erzeugten Cluster werden in einer Baumstruktur gespeichert, inklusive des Ähnlichkeitswertes, bei dem sie aus zwei der bisherigen Clustern erzeugt wurden. Diese beiden werden als Kinder des Clusters festgehalten. Außerdem ist für jeden Cluster bekannt, welche Elemente er enthält und welches Level er hat, also wie viele Hierarchiestufen sich unter ihm befinden. Um die Durchführung zu beschleunigen, können alle Resultate der Scanpath-Vergleiche untereinander in einer Matrix gespeichert werden. Während des Clusterings können diese dann dort nachgeschaut werden, wodurch die Notwendigkeit einer zeitaufwendigen erneuten Berechnung entfällt.

6.7 Visualisierung der Ergebnisse

In dieser Arbeit wurden zwei grundlegend verschiedene Arten von Visualisierungen genutzt. Eine Tabelle für eine vollständige, aber schwer überschaubare Anzeige aller berechneten Ergebnisse sowie eine vereinfachte Darstellung der sich daraus ergebenden Hierarchie als Dendrogramm und interaktive Baumansicht.

6.7.1 Vergleichstabelle

Eine Tabellenansicht ermöglicht die Betrachtung aller Vergleichswerte auf einen Blick. Bei einer großen Datenmenge leidet darunter jedoch der Überblick. Um diese Schwäche auszugleichen, wurde eine Einfärbung in Abhängigkeit von der relativen Größe eines Wertes verwendet. Die Farben werden in der prototypischen Implementierung nach der selben Vorgehensweise gewählt, wie sie in Abschnitt 6.4.1 beschrieben wurde.

Abb. 6.5 zeigt ein Beispiel für diese Visualisierung. Bei den hier gezeigten Daten handelt es sich um 30 generierte Scanpaths. Diese wurden ausgehend von drei zuvor zufällig erzeugten Pfaden durch eine ebenfalls zufällige Variation erzeugt. Ohne weitere Vorverarbeitung wurden sie mit dem Algorithmus von Needleman und Wunsch verglichen, wobei die Ergebnisse normalisiert wurden. Die Tabelle zeigt eine relativ hohe Ähnlichkeit innerhalb der entsprechenden Gruppen. Zu sehen ist das anhand der blauen Einfärbung der entsprechenden Tabellenfelder. Diese nimmt jedoch mit steigender Standardabweichung der für die Erzeugung der Varianten verwendeten Normalverteilung ab. Trotzdem ist die Ähnlichkeit innerhalb der Gruppen höher als zwischen ihnen.

6.7.2 Dendrogramm und Baumansicht

Im Gegensatz zur Vergleichstabelle werden bei einem Dendrogramm die Ähnlichkeiten nicht explizit angezeigt. Stattdessen wird die Hierarchie als Baumvisualisierung gezeichnet. In Abb. 6.6 werden die selben Ergebnisse visualisiert, die zuvor in Abb. 6.5 in einer Tabelle gezeigt wurden. Veranlasst

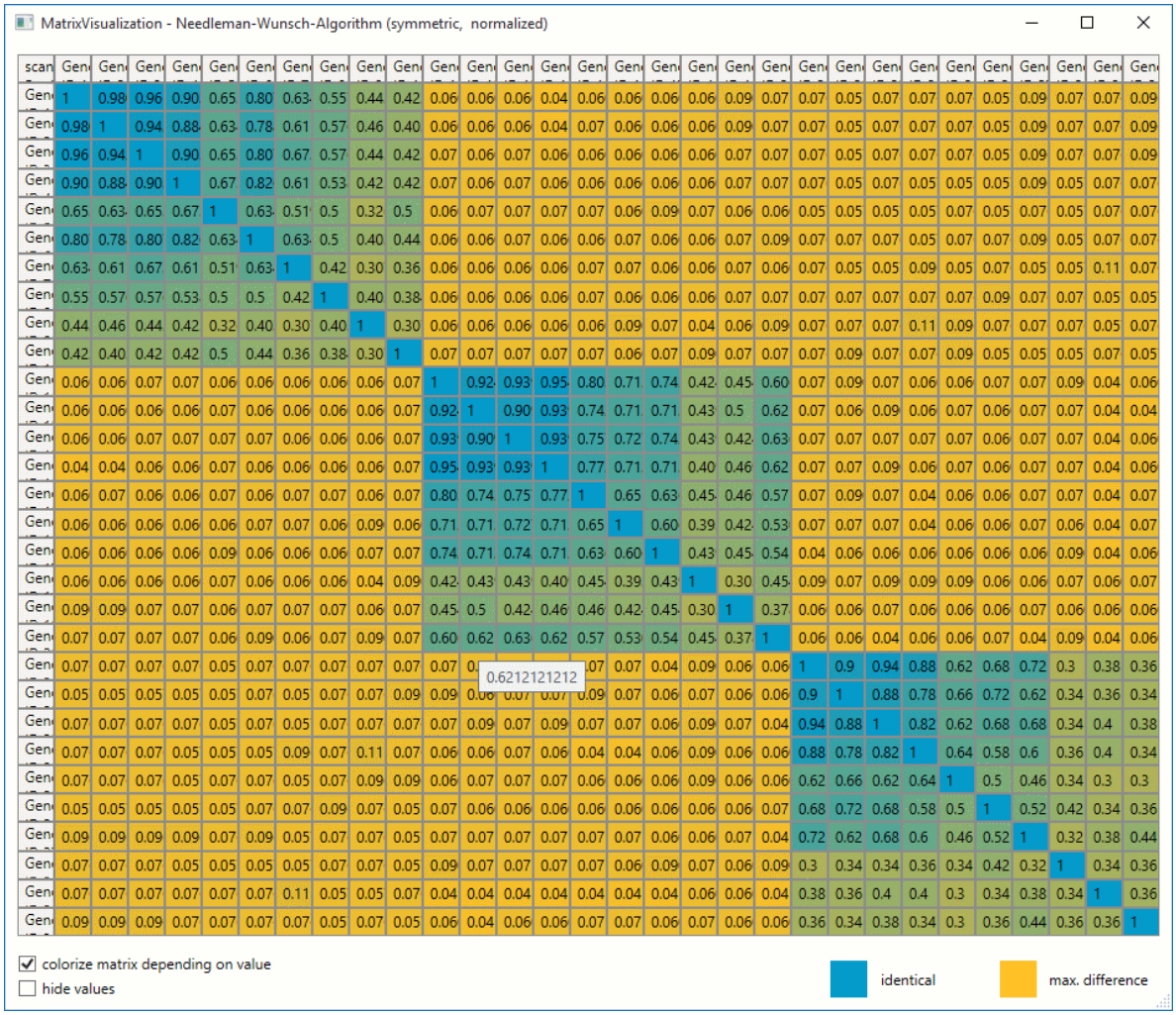


Abbildung 6.5: Visualisierung der Ähnlichkeiten in einem Satz von Scanpaths durch eine Tabelle. Die Scanpaths wurden alphanumerisch nach dem Namen des Probanden geordnet. Je nach Wert wurden die Tabellenfelder eingefärbt, Blau steht hierbei für eine hohe, gelb für eine geringe Ähnlichkeit. Ein Tooltip zeigt die ungekürzten Werte oder Beschreibungen, die aufgrund der Anzahl der zu zeigenden Elemente keinen Platz finden. Optional können die Werte auch ausgeblendet oder ohne Einfärbung angezeigt werden. Ein Klick auf einen Wert öffnet ein Fenster, dass die beiden dazugehörigen Scanpaths anzeigt.

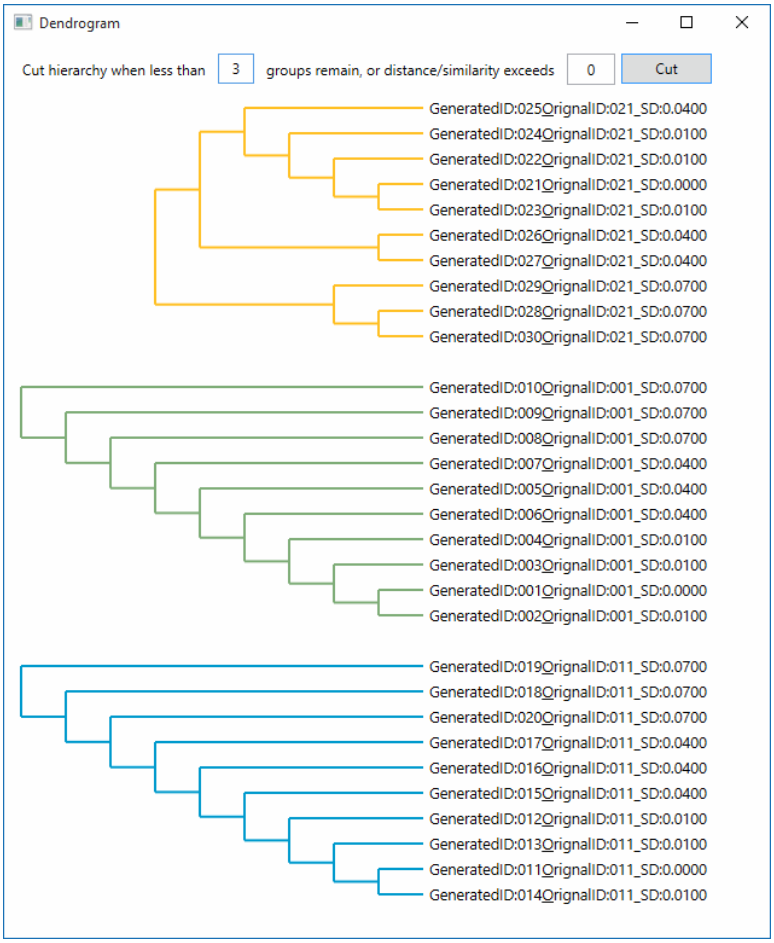


Abbildung 6.6: Visualisierung der Ähnlichkeiten durch ein Dendrogramm. Die Hierarchie wurde vom Benutzer in drei Gruppen geschnitten. Je nach Gruppe wählt die Visualisierung automatisch eine Farbe. Die Gruppierung der randomisiert generierten Scanpaths war dann erfolgreich, wenn jede Gruppe nur Scanpaths enthält, die Varianten des selben Originals sind. Alle Pfade wurden in diesem Fall korrekt gruppiert, wie an der im Namen vermerkten ID des Originalpfades zu erkennen ist.

durch eine Benutzereingabe wurde die Hierarchie auf einem Niveau abgeschnitten, sodass drei Gruppen entstanden sind. Dabei ist ersichtlich, dass alle Scanpaths korrekt ihrem Original zugeordnet wurden. Wie erwartet wurden außerdem Varianten mit ähnlicher Standardabweichung als zueinander ähnlicher betrachtet. In diesem Fall war die verwendete Metrik unter den gewählten Parametern also für den Vergleich geeignet.

Für eine bessere Lesbarkeit der Darstellung werden auch hier Farben verwendet, die analog zur Vergleichstabelle gewählt werden. Die Größe der Graphik und die Textgröße skalieren automatisch zur Anzahl der zu zeigenden Scanpaths und zur Fenstergröße.

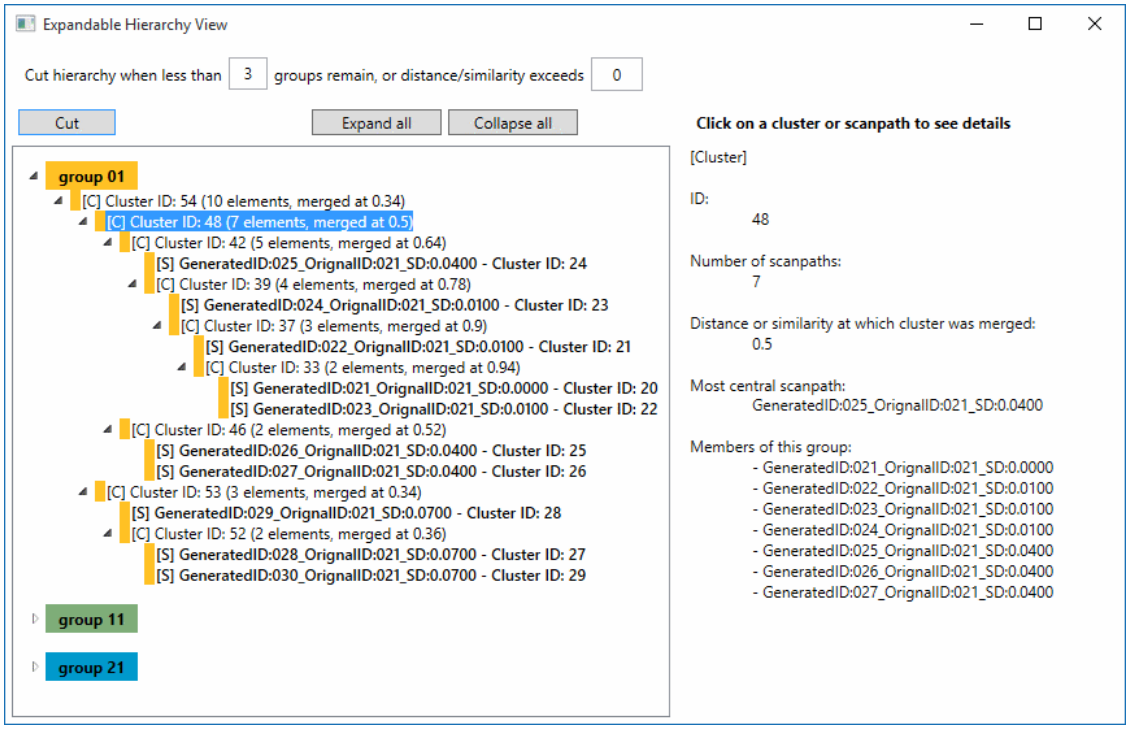


Abbildung 6.7: Visualisierung der Scanpath-Ähnlichkeiten durch eine Baumansicht. Die gezeigten Daten werden zur Dendrogramm-Visualisierung synchron gehalten, ebenso die Einfärbung der Gruppen. Jeder Cluster kann erweitert oder zusammengefasst werden. Bei einem Klick auf einen Scanpath oder Cluster werden dessen Informationen angezeigt, wie beispielsweise die in einem Cluster enthaltenen Scanpaths.

Die Baumansicht zeigt die Daten synchron zum Dendrogramm an. So werden in Abb. 6.7 die selben Daten gezeigt, wie in Abb. 6.6. Werden in einem der beiden Fenster die Parameter zur Gruppenbildung geändert, so wird diese Änderung auch im anderen Fenster übernommen. Für die Einfärbung der Gruppen werden die selben Farben verwendet. Der Benutzer hat die Möglichkeit, Cluster einzuklappen, um nur momentan relevante Daten anzeigen zu lassen. Über entsprechende Buttons können alle Elemente auf einmal auf- oder zugeklappt werden. Zusätzlich kann mit einem Klick auf einen Cluster oder einen Scanpath eine Auflistung der dazugehörigen Informationen veranlasst werden. Diese zeigt das Programm dann in einem abgesonderten Bereich an. Hier werden für Cluster unter anderem die enthaltenen Scanpaths aufgelistet und der Wert genannt, bei dem die beiden Kind-Cluster vereinigt wurden.

7 Demonstration und Evaluation

Das Ziel dieser Arbeit ist der Vergleich verschiedener Scanpath-Vergleichsmetriken. In diesem Kapitel werden die durchgeführten Versuche vorgestellt, mit denen die im Konzept ausgewählten Metriken unter verschiedenen Konfigurationen verglichen werden. Das betrifft insbesondere die Parameter, sodass beispielsweise unterschiedliche Kosten für dier Levenshtein-Distanz angesetzt werden. Die Longest Common Subsequence unterstützt keine eigenen Parameter und wird lediglich durch die Vorverarbeitung der Scanpaths und die im dritten Versuch verwendeten Gewichtungsfunktionen beeinflusst.

Außerdem kommen bei den Versuchen Scanpaths zum Einsatz, die auf unterschiedliche Weise entstanden sind. Sie werden weiterhin je nach Versuch einer Vorverarbeitung unterzogen, bei der ebenfalls verschiedene Einstellungen vorgenommen werden, um eine optimale Konfiguration zu finden. Die Versuche werden zunehmend realitätsnäher und damit komplexer. Während im ersten Versuch künstlich erstellte Daten verarbeitet werden, findet in den beiden anderen Versuchen eine Analyse realer Eye-Tracking-Daten statt. Diese sind im zweiten Versuch durch Muster im Stimulus beeinflusst, im dritten Versuch nur durch verschiedene Aufgaben der Probenden.

Bei allen Versuchen sind aufgrund der Generierung oder Beeinflussung bestimmte Erwartungen an die Ähnlichkeit zwischen den Scanpaths vorhanden. Durch diese Erwartungen ist für jeden Scanpath im Voraus bekannt, zu welcher Gruppe er gehört. Nach dem Vergleich der Scanpaths wird ein Clustering zur Erzeugung der selben Anzahl von Gruppen aus den Daten durchgeführt. So lässt sich anhand der Gruppenzuordnungen erkennen, welche Testkonfiguration die vorhandenen Erwartungen am ehesten erfüllt und damit am geeignetsten erscheint.

7.1 Versuch 1: Randomisiert generierte Testdaten

Um möglichst große Datenmengen und damit eine höhere Genauigkeit der Ergebnisse zu erhalten, wurden im ersten Versuch automatisch generierte Testdaten verwendet. Die folgenden Abschnitte erläutern die Vorgehensweise zur Erzeugung dieser Daten. Außerdem wird auf die Konfiguration der Metriken und die Durchführung mehrerer Tests eingegangen. Abschließend werden die Ergebnisse gezeigt und diskutiert.

7.1.1 Erzeugung der Testdaten

Die Testdaten wurden wie in Abschnitt 5.7.1 erläutert generiert. Es wurden dabei zufällig gleichverteilt Punkte auf einem hypothetischen zweidimensionalen Stimulus platziert, um Muster-Scanpaths zu erhalten. Diese wurden dann Punkt für Punkt zufällig verändert, indem durch eine Normalverteilung

ermittelte Wert auf die Koordinaten der Punkte addiert wurden. Dadurch entstanden Varianten mit einer durch die Standardabweichung der Normalverteilung kontrollierbarer Ähnlichkeit zum Original.

Für die Länge der Scanpaths wurde ein Wert von 60 Fixationspunkten gewählt, was mehrere Gründe hat. Zunächst wird auf diese Weise verhindert, dass Unterschiede in der Länge einen Einfluss auf die Berechnung der Ähnlichkeit haben. Außerdem werden aufgrund der zufälligen Scanpath-Generierung längere Pfade unterschiedlicher zueinander, was eine Unterscheidung der Gruppen für die Metriken zu einfach machen würde. Ein weiterer Grund ist die benötigte Rechenzeit, da etwa der Needleman-Wunsch-Algorithmus eine Komplexität von $\mathcal{O}(max(m,n)^3)$ aufweist, wobei n und m für die Längen der verglichenen Scanpaths stehen. Zuletzt passen die Pfade mit der gewählten Länge zu den in Versuch 2 (Abschnitt 7.2) aufgezeichneten Pfaden, die ebenfalls um die 60 Fixationen enthielten.

Der Versuch bestand aus neun getrennten Tests. Für jeden der Tests wurden zunächst vier verschiedene Pfade generiert. Von diesen wurden dann wiederum je 100 Varianten erstellt, wobei die Standardabweichung der Normalverteilung zwischen den Tests in Schritten von 0,1 zwischen 0,1 und 0,5 verändert wurde. Dies entspricht in etwa der bei MultiMatch verwendeten Vorgehensweise (siehe Abschnitt 4.2.3). Insgesamt entstanden bei dieser Generierung 400 Scanpaths für jeden Test.

Bis zu diesem Punkt bestanden die Scanpaths aus Punkten auf einem zweidimensionalen Stimulus. Diese Punkte wurden im nächsten Schritt gitterförmigen AOIs zugeordnet, wobei das Gitter aus 40 AOIs in fünf Reihen und acht Spalten bestand. Die Werte wurden gewählt, da dieselben AOIs auch im nächsten Versuch (siehe Abschnitt 7.2) Verwendung finden sollten, in welchem sie aufgrund des Monitorformates und der gewählten Gitterform eine annähernd quadratische Form hatten.

Nach dieser AOI-Zuordnung bestanden die Scanpaths nun nicht mehr aus Fixationspunkten, sondern aus Zahlen für die jeweils zur Fixation gehörenden AOIs. Dadurch können die Scanpaths durch die Metriken verglichen werden, indem die Zahlen wie Buchstaben in einem String behandelt werden. AOI-abhängige Parameter werden zudem während der Berechnung anhand der Zahlen in einer Tabelle nachgeschaut. Auf eine Vorverarbeitung wurde in diesem Versuch verzichtet. Temporal Binning konnte aufgrund der mangelnden zeitlichen Komponente nicht angewandt werden. Einige Test hatten zudem gezeigt, dass Filterung und Zusammenfassung von AOI-Aufenthalten einen vernachlässigbaren Einfluss auf die Ergebnisse haben, da nur wenige hintereinanderliegende Fixationen in der selben AOI landeten. Dies liegt an der verwendeten Methode zur Generierung der Pfade, bei der die Fixationspunkte zufällig gleichverteilt auf dem Stimulus gesetzt wurden. Eine Filterung würde aus ähnlichen Gründen keine Verbesserung des Vergleichs bewirken, sondern bei einem Grenzwert von Zwei oder höher große Teile der Scanpaths entfernen. Es wurde zudem beobachtet, dass eine Kompression der Scanpaths in beinahe jedem Fall eine Verschlechterung des Ergebnisses zur Folge hatte. Aufgrund dieser Sachverhalte wurden die Vorverarbeitungsschritte erst in den folgenden beiden Versuchen angewandt, die in den Abschnitten 7.2 und 7.3 behandelt werden.

7.1.2 Konfiguration der Metriken

In jedem Test wurden die Scanpaths mit jeder Metrik verglichen. Dabei kamen bei Damerau-Levenshtein und Needleman-Wunsch in getrennten Durchläufen einmal die Standardparameter und einmal AOI-abhängige Parameter zum Einsatz. Eine Funktion zur Gewichtung der Parameter

Metrik / Standardabweichung	0,1	0,2	0,3	0,4	0,5
Longest Common Subsequence	52,00	33,75	29,75	29,00	29,75
Damerau-Levenshtein	100,00	62,00	41,50	30,25	29,75
Damerau-Levenshtein (AOI-abh.)	40,75	35,25	28,75	28,50	29,25
Needleman-Wunsch	100,00	41,00	31,50	28,50	30,25
Needleman-Wunsch (AOI-abh.)	100,00	84,25	36,50	28,50	30,50

Tabelle 7.1: Ergebnisse des ersten Versuchs. In dieser Tabelle ist der prozentuale Anteil der korrekt nach Originalpfaden gruppierten Scanpaths pro Metrik und Test vermerkt. Die zu vergleichenden Daten waren randomisiert generierte Scanpaths. Die Algorithmen von Damerau-Levenshtein und Needleman-Wunsch wurden je einmal mit und einmal ohne AOI-abhängige Parameter verwendet, die Longest Common Subsequence unterstützt diese Option nicht.

über den zeitlichen Verlauf der Scanpaths wurde nicht verwendet, da alle Pfade die selbe Länge haben und konzeptbedingt an jeder Stelle erwartet gleich stark verrauscht sind. Daher gibt es keine Abschnitte, deren Ähnlichkeit im Vergleich zu anderen relevanter sind. Auch eine Normalisierung der Ergebnisse ist aufgrund der identischen Längen nicht notwendig.

7.1.3 Auswertung und Ergebnisse

Um das Ergebnis bewerten zu können, wurde die Anzahl der resultierenden Gruppen entsprechend den bei der zufälligen Erzeugung der Scanpaths verwendeten originalen Pfaden gewählt. Diesen Gruppen sollten nun möglichst alle ihre Varianten zugeordnet werden. Um die Güte der Zuordnung zu berechnen, wurde der Anteil der korrekt zugeordneten Scanpaths an der Gesamtzahl der Pfade betrachtet. Die Ergebnisse aufgeteilt nach Test und Metrik sind in Tabelle 7.1 zu sehen.

Erwartungsgemäß sinkt die Qualität der Ergebnisse mit zunehmender Standardabweichung. Die Longest Common Subsequence weist dabei schon zu Beginn sehr schwache Ergebnisse auf. Nur bei schwach verrauschten Daten liegt sie deutlich über einem Wert von 25 Prozent korrekter Zuordnungen, der erwartet durch Raten der Zuordnung zu den vier Gruppen zu erreichen wäre. In diesem Versuch konnte die Levenshtein-Distanz nicht von den AOI-abhängigen Parametern profitieren. Der Algorithmus von Needleman und Wunsch hingegen erreicht mit AOI-abhängigen Punktzahlen bei einer Standardabweichung von 0,2 einen mehr als doppelt so guten Wert. Dies liegt möglicherweise an einer ungünstigen Wahl des Bereiches liegen, in den die Ersetzungskosten der Levenshtein-Distanz nach der Normalisierung zur Stimulusdiagonalen skaliert wurden.

7.2 Versuch 2: Daten aus einer Benutzerstudie

Die im ersten Versuch verwendeten Daten haben nur geringe Ähnlichkeit mit den bei einer echten Eye-Tracking-Studie entstehenden Daten. Ein zweiter Versuch mit echten Probanden soll einen Kompromiss zwischen Datenmenge und Realismus darstellen. Dabei wurden erneut Scanpaths wie

im ersten Versuch generiert. Deren Pseudofixationen wurden dann auf einem Bildschirm angezeigt und die Blicke der Probanden aufgezeichnet, wodurch sich Muster in den real aufgezeichneten Daten vorgeben ließen.

7.2.1 Teilnehmer

An der Studie nahmen sieben Testpersonen im Alter von 20 bis 26 Jahren teil, das Durchschnittsalter betrug 22,3 Jahre. Fünf davon gaben ihr Geschlecht als männlich an, eine als weiblich, eine wollte ihr Geschlecht nicht angeben. Zwei der Personen trugen eine Sehhilfe. Alle Teilnehmer mussten einen Test für Sehstärke und Farbenblindheit (Ishihara-Test) durchführen, um die allgemeine Eignung für eine Eye-Tracking Studie festzustellen. Da alle Personen eine korrekte oder korrigierte Sicht hatten, konnte jede mit der Durchführung der Studie fortfahren. Die Studie dauerte insgesamt weniger als eine halbe Stunde, entschädigt wurden die Teilnehmer, wie im Voraus angekündigt, mit einem Sachpreis von geringem Wert.

7.2.2 Gerät

Der in der Benutzerstudie verwendete Eye-Tracker ist der Bildschirm-basierte Tobii T60XL (siehe auch Abb. 2.3). Der Stimulus wurde im Vollbild bei einer Bildschirmauflösung von 1920×1200 Pixeln und einer Diagonalen von 24 Zoll angezeigt. Die Kamera des Eye-Trackers nimmt die fixierte Position mit 60 Samples pro Sekunde auf. Die Genauigkeit unter idealen Bedingungen beträgt laut Hersteller 0.4 Grad bei binokularer Messung und einem Abstand zwischen Auge und Bildschirm von 65 Zentimetern. Die aufgenommenen Blickpunkte wurden mit dem Tobii Fixation Filter zu Fixationen zusammengefasst. Dabei wurden die Standardeinstellungen verwendet, insbesondere ein Abstandsgrenzwert von 35 Pixeln und ein Geschwindigkeitsgrenzwert von 35 Pixeln pro Sample.

7.2.3 Stimulus

Als Stimulus wurden drei Muster-Scanpaths verwendet, die mit der in Abschnitt 5.7.1 beschriebenen Methode erzeugt wurden. Abb. 7.1 zeigt die Darstellung des Stimulus, wie ihn die Probanden sahen. Es wurden drei verschiedene Pfade generiert, von denen zwei aus jeweils 27 und einer aus 29 Kreisen bestanden. Für eine optimale Aufzeichnung und spätere Zuordnung der Fixationspositionen wurde der Stimulus im Vollbild angezeigt. Die Kreise hatten einen Durchmesser von 20 Pixeln, um eine gute Sichtbarkeit zu erreichen. Durch die geringen Ausmaße und einen Punkt in der Mitte der Kreise wurde ein genaues Fokussieren erleichtert.

7.2.4 Durchführung

Nach den üblichen Formalitäten wurden die Probanden über ihre Aufgabe informiert. Diese bestand darin, den aktuell auf dem Bildschirm zu sehenden Kreis so genau wie möglich zu fokussieren.

Abbildung 7.1: Der für die Benutzerstudie verwendete Stimulus, wie ihn die Teilnehmer sahen. Nacheinander wurden einem vorher generierten Muster-Scanpath entsprechend Kreise im Abstand von je einer Sekunde angezeigt. Der graue Rand dient zur Verminderung der Auswirkung von Ungenauigkeiten des Eye-Trackers im Randbereich des Bildschirms. Eine vollständige Darstellung eines Muster-Scanpaths mit allen Kreisen ist in Abb. 7.2 zu sehen.

Vor der Durchführung der Aufnahmen wurde für jeden Teilnehmer die korrekte Position der Augen überprüft und einmalig eine 9-Punkte-Kalibrierung des Eye-Trackers vorgenommen, um eine möglichst genaue Messung sicherzustellen. Danach wurde für jeden gezeigten Muster-Scanpath des Stimulus eine eigene Eye-Tracking-Aufnahme vorgenommen. Jeder Proband bekam zuerst jeden der drei Pfade einmal zu sehen, danach wurden die Pfade in der gleichen Reihenfolge erneut gezeigt. Daraus resultierten sechs Aufnahmen pro Versuchsperson, also insgesamt 42 Datensätze.

Bevor der Stimulus selbst angezeigt wurde, musste der Proband durch Betätigen der Enter-Taste anzeigen, dass er für die Bearbeitung der Aufgabe bereit ist. Dieses Event wurde aufgezeichnet und erlaubte eine spätere Aussortierung von nicht relevanten Daten. Nach dem Tastendruck begann mit einer Sekunde Verzögerung die Anzeige des eigentlichen Stimulus. Die Kreise der Muster-Scanpaths wurden dabei für je eine Sekunde angezeigt, woraufhin unmittelbar der nächste folgte. Auf diese Weise sollte den Probanden keine Zeit gelassen werden, etwas anderes als den aktuellen Kreis zu fokussieren. Außerdem vereinfacht sich die Auswertung, da der Zeitpunkt des Erscheinens jedes Kreises bekannt ist. Jeder Durchlauf dauert aufgrund dieser Wartezeit knapp 30 Sekunden.

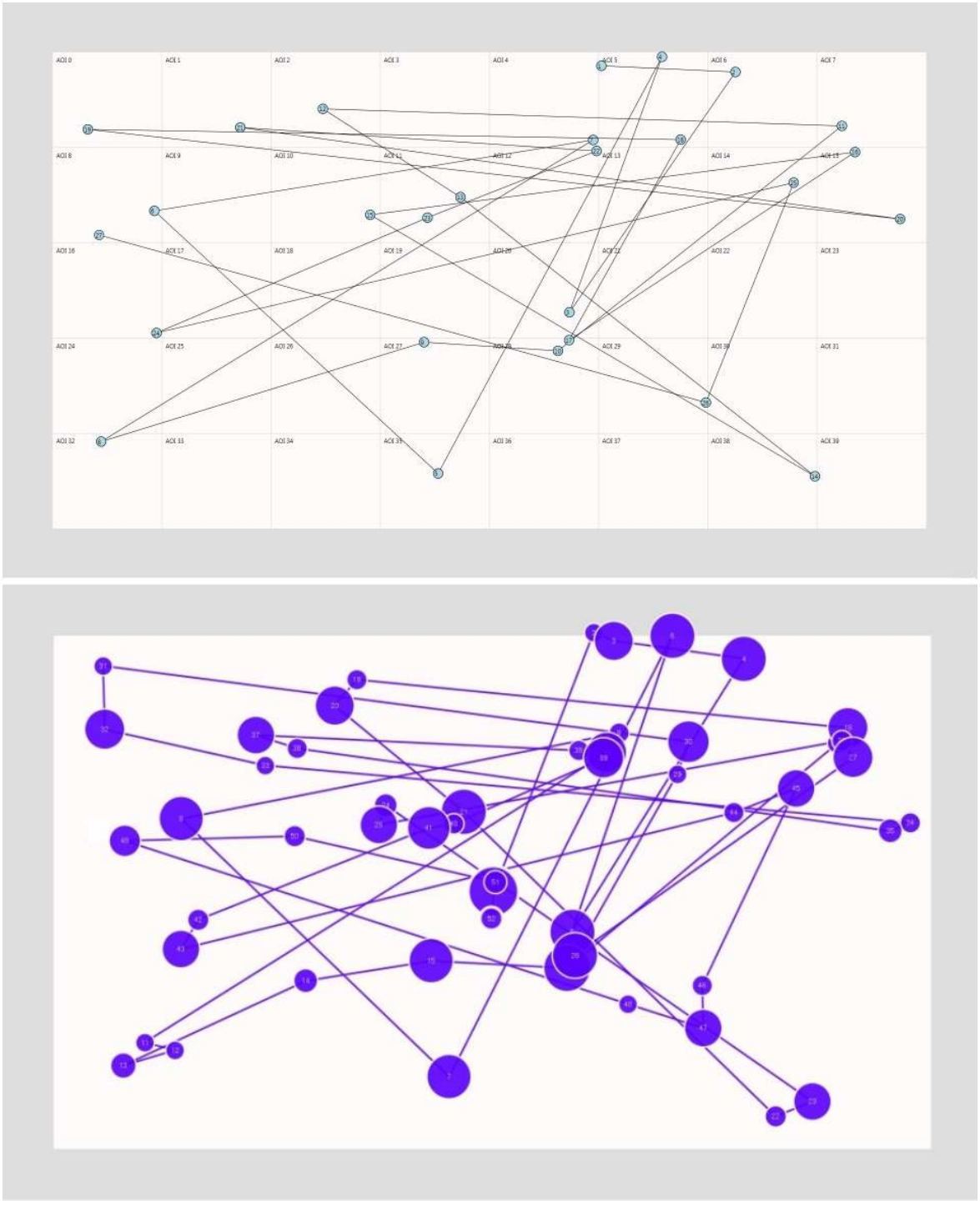


Abbildung 7.2: Vergleich von Muster-Scanpath (oben) und aufgezeichneten Daten (unten). Im oberen Bild sind die verwendeten AOIs zu sehen. Die violetten Kreise im unteren Bild stellen Fixationen dar, die Größe entspricht der Fixationsdauer. Das untere Bild wurde mit der Tobii Studio Software erzeugt.

7.2.5 Auswertung

Abb. 7.2 zeigt vergleichsweise, wie ein aufgezeichneter Scanpath zu einem gewählten Muster-Scanpath aussehen kann. Wo sich im Stimulus ein Kreis befand, wurde teils korrekterweise eine etwa eine Sekunde dauernde Fixation erkannt. Manchmal wurden die Blickpunkte zu mehreren kurzen Fixationen zusammengefasst. Lagen zwei aufeinanderfolgende Kreise weit von einander entfernt, kam es teilweise zu kürzeren Fixationen zwischen diesen oder hinter dem neuen Kreis. Deren negative Auswirkung auf den Scanpath-Vergleich kann durch Temporal Binning und Filterung kurzer AOI-Aufenthalte verringert werden.

Um Störungen durch nicht relevante Daten zu vermindern, wurden Fixationen aus den Aufzeichnungen entfernt, die vor oder nach der Anzeige des Stimulus stattgefunden hatten. Dabei wurden alle Fixationen aussortiert, deren Beginn vor dem Betätigen der Enter-Taste durch den Probanden lag. Zusätzlich wurden auch Fixationen entfernt, die mehr als 30 Sekunden nach Beginn des Stimulus endeten. Der Grund dafür ist, dass die letzte während dem Stimulus aufgezeichnete Fixation stets auf der Benachrichtigung über das Ende desselben in der Mitte des Bildschirms lag. Auf diese Weise wurden im Schnitt drei bis acht für den Versuch nicht benötigte Fixationen entfernt, was etwa zehn Prozent der ursprünglichen Daten ausmachte.

Für verschiedene Kombinationen von Vorverarbeitungsparametern wurde je eine eigene Scanpathkonfiguration erstellt. Weiterhin wurden je nach Metrik ebenfalls verschiedene Parameter in unterschiedlichen Kombinationen gewählt und dafür und für jede Scanpathkonfiguration eine Testkonfiguration erzeugt. Mit den Testkonfigurationen wurde jeweils der Vergleich und die Gruppierung aller Scanpaths vorgenommen, indem die in ihr gespeicherte Metrik unter den ebenfalls enthaltenen Parameterwerten zum Vergleich aller Scanpaths eingesetzt wurde.

Die binären Vorverarbeitungsparameter *Kompression* und *Zusammenfassung* wurden unabhängig voneinander je einmal aktiviert und deaktiviert verwendet. *Temporal Binning* und *Filterung* wurden ebenfalls einmal deaktiviert und aktiviert. Für die aktivierte Option wurden je drei Schritte mit verschiedenen Grenzwerten zwischen 45 und 55 Millisekunden beziehungsweise einer Filtergröße von 1 bis 3 Zeichen verwendet. Es wurden alle Metriken jeweils mit Standard- und AOI-abhängigen Parametern eingesetzt, insofern diese unterstützt wurden. Dabei wurden die selben Parameter wie in Versuch 1 verwendet. Aufgrund der unterschiedlich langen Scanpaths wurde in jedem Fall eine Normalisierung der Ergebnisse angewandt. Auf eine Gewichtung der Metrikparameter mittels zeitabhängiger Funktionen wurde auch in diesem Versuch verzichtet, da sich die Länge der Pfade nicht wesentlich unterschied und keine Abschnitte mit besonderer Relevanz vorkamen. Insgesamt entstanden anhand dieser Parameterbereiche 64 Scanpathkonfigurationen und 256 Testkonfigurationen. Der Prototyp führte alle Testkonfigurationen parallel aus, die Berechnung der aufwändigsten Konfiguration und damit die Laufzeit des Tests betrug knapp sieben Stunden auf einem modernen Desktop-PC mit Intel Xeon CPU (8×3,40 GHz).

7.2.6 Ergebnisse

Aus den ermittelten Erfolgsquoten der 256 Testkonfigurationen wurden Durchschnittswerte für verschiedene Parameterzustände berechnet. Diese sind in Tabelle 7.2 zu sehen.

Parameter, Metrik	angewandt	nicht angewandt
Temporal Binning	68,48	57,44
Filter	65,03	67,78
Zusammenfassung	64,92	66,52
Kompression	45,88	77,62
Parameter AOI-abh.	73,85	63,01
LCS	68,56	
Damerau-Levenshtein	63,89	
Needleman-Wunsch	65,65	
Schnitt insgesamt	65,72	

Tabelle 7.2: Durchschnittliche Ergebnisse des zweiten Versuchs. In dieser Tabelle sind die Durchschnitte der prozentualen Erfolgsquote bei der Zuordnung der Scanpaths zur korrekten Gruppe aufgeführt, je nach Aktivierungsstatus eines Parameters oder einer Metrik. In diesem Versuch wurden die Daten aus einer Eye-Tracking Studie mit vorgegebenen Pfaden analysiert. Ein hoher Wert deutet auf eine positive Auswirkung der entsprechenden Option hin.

Zunächst fiel bei der Analyse der Ergebnisse auf, dass sich eine Verwendung der Longest Common Subsequence im Vergleich zum ersten Versuch als unerwartet erfolgreich bei der korrekten Zuordnung erwies. Dies könnte ein Hinweis sein, dass die bei der Studie aufgezeichneten Daten Ähnlichkeitsgruppen bildeten, die sehr unterschiedlich zueinander sind, bei gleichzeitig hoher Ähnlichkeit der in den Gruppen enthaltenen Scanpaths. Dadurch wurde möglicherweise der Vergleich für den LCS-Algorithmus einfacher, während die anderen beiden Metriken ihre Vorteile nicht ausspielen konnten.

Temporal Binning führt bei aufgezeichneten Daten zu einer deutlichen Verbesserung der Ergebnisse. In diesem Versuchsdurchlauf verbesserte sich das Resultat um über 11 Prozentpunkte, was im Vergleich zu den anderen Parametern ein relativ hoher Wert ist. Eine Filterung scheint durch die Reduktion der Daten einen negativen Einfluss zu haben. Im Schnitt wurden die Ergebnisse zudem besser, wenn auf eine Zusammenfassung von AOI-Aufenthalten verzichtet wurde. Das liegt daran, dass durch diesen Schritt Unterschiede in den Scanpaths verloren gehen können.

Es zeichnete weiterhin sich ab, dass sich das Ergebnis bei aktivierter Kompression grundsätzlich verschlechtert. Dabei wurde die Longest Common Subsequence am wenigsten beeinflusst. Trotz Kompression konnte eine Testkonfiguration eine Erfolgsquote von 73 Prozent erreichen. Der nächst beste Fall mit aktivierter Kompression war eine Konfiguration mit Needleman-Wunsch, die eine Quote von 59,52 Prozent hatte. Ein Grund dafür ist womöglich, dass ein komprimierter Scanpath die Verwendung von AOI-abhängigen Parametern ausschließt.

Die Metriken mit Unterstützung für AOI-abhängige Parameter konnten von diesen profitieren. So verbesserte sich das Ergebnis im Schnitt um 10,84 Prozentpunkte, wenn Informationen über die Ähnlichkeit der AOIs in die Berechnung einbezogen wurden.

Parameter, Metrik	angewandt	nicht angewandt
Temporal Binning	85,28	61,55
Filter	77,98	83,45
Parameter AOI-abh.	73,36	83,33
LCS	93,45	
Damerau-Levenshtein	73,59	
Needleman-Wunsch	78,05	
Schnitt insgesamt	79,35	

Tabelle 7.3: Durchschnittliche Ergebnisse des zweiten Durchlaufs in Prozent.

Aufgrund dieser Beobachtungen wurde der Versuch erneut ohne Kompression und Zusammenfassung durchgeführt. Dadurch soll untersucht werden, wie sich die Metriken ohne diese Vorverarbeitungsschritte verhalten. Tabelle 7.3 enthält die dabei gewonnenen Ergebnisse.

Der positive Effekt von Temporal Binning zeichnet sich nun noch deutlicher ab, ebenso die negative Auswirkung der Filterung. Der Grund für die Verschlechterung des Ergebnisses durch die Verwendung AOI-abhängiger Parameter sind die hohe Erfolgsquote der LCS und die schlechten Ergebnisse der AOI-abhängigen Levenshtein-Distanz, die auch ohne Kompression und Filterung im Vergleich zur Standardimplementierung versagt.

7.3 Versuch 3: Evaluation anhand externer Daten

Um eine Evaluation der Metriken und Vorverarbeitung mit echten Daten durchzuführen, wurden im dritten Versuch die Aufzeichnungen einer Eye-Tracking-Studie [20] verwendet, die am Institut für Visualisierung und Interaktive Systeme der Universität Stuttgart durchgeführt wurde. Das Ziel dieser Studie war die Erzeugung von Standarddaten zum Testen von Eye-Tracking-Analysewerkzeugen und -Visualisierungen. Analog zu einem Textkorpus oder Standardbilddaten in der Text- oder Bildverarbeitung kann so jede entwickelte Technik anhand der selben Daten evaluiert werden, um vergleichbare Ergebnisse zu erhalten. Bei diesem Versuch soll festgestellt werden, wie sich die Metriken unter Verwendung von semantikbasierten Parametern verhalten.

7.3.1 Durchführung

Zur Evaluation dieser Arbeit wurde der Datensatz *S10*[1] verwendet. Dabei handelt es sich um eine Aufzeichnung der Augenbewegungen von 25 Probanden, die ein Video betrachteten, in dem Personen Taschen durch den sichtbaren Bereich tragen. Die Aufgabe bestand darin, eine bestimmte Tasche zu finden. Dabei gab es zwei Gruppen von Probanden mit verschiedenen Suchzielen. Nach der Aufnahme

[1] www.visus.uni-stuttgart.de/index.php?id=2345

wurden dynamische AOIs aus den Daten erzeugt und die Fixationen der Probanden diesen AOIs zugeordnet, wodurch sich AOI-Label-Scanpaths erstellen ließen.

Die Ähnlichkeit der AOIs kann aufgrund der sich ändernden Position von der für diese Arbeit erstellten Implementierung nicht distanzbasiert berechnet werden. Allerdings kann von einer semantischen Ähnlichkeit ausgegangen werden, da die AOIs zu Objekten gehören die verschiedenen Gruppen zugeordnet werden können. In diesem Fall handelt es sich dabei getrennt nach Eigenschaft um folgende Gruppen: Nach Art des Objekts ergeben sich je eine Gruppe für Taschen und eine für Personen. Differenziert nach Farbe entstehen Gruppen für rote, gelbe, blaue, rot-weiße, braune und farblose Objekte.

Abhängig von der Gruppenzugehörigkeit und damit der Semantik können nun die Parameter der Metriken festgelegt werden, wie in Abb. 7.3 zu sehen ist. Das Ziel dabei ist, dass zwei Scanpaths als lokal ähnlicher gelten, wenn die gerade verglichenen AOIs in einer oder mehreren gemeinsamen Gruppen sind. Zum Beispiel gilt eine gelbe Tasche als ähnlicher zu einer anderen gelben Tasche als zu einer roten oder zu einer Person.

Die Scanpaths der Probanden lagen als Datei vor, in der die Fixationen bereits den AOIs zugeordnet und durch je eine Zahl repräsentiert wurden. Aufgrund des Samplings der Daten für jeden Frame des Videos entstanden 24 Werte für jede Sekunde der Aufzeichnung. Das entspricht einem Temporal Binning mit einer Zeitspanne von etwa 41,7 Millisekunden. Da dieser Umstand jedoch zu einer Scanpathlänge von circa 2500 bis 3000 Zeichen führte, mussten die Daten erst vereinfacht werden, um eine Auswertung in wenigen Minuten statt einigen Stunden ermöglichen. Hierfür wurden die AOI-Aufenthalte zu je einem Zeichen zusammengefasst. Eine Filterung kurzer AOI-Aufenthalte führte im Schnitt zu keiner Änderung des Ergebnisses, weshalb auf diese Option verzichtet wurde, um die Anzahl der zu berechnenden Testkonfigurationen zu verringern.

In diesem Versuch wurden die Metrikparameter durch eine zeitabhängige Funktion gewichtet. Dabei kam eine abschnittsweise definierte Funktion zum Einsatz, die zu den Zeiten ein höheres Gewicht zurückgibt, bei denen im Videostimulus eines oder mehrere der Suchziele zu sehen waren:

$$f_{a,b}(t) = \begin{cases} a & \text{zur Zeit t ist kein Ziel zu sehen} \\ b & \text{zur Zeit t ist mindestens ein Ziel zu sehen} \end{cases} \tag{7.1}$$

Durch diese Gewichtung haben diese relevanten Ausschnitte einen höheren Einfluss auf das Ergebnis der Berechnung. Da im Voraus nicht bekannt ist, welche Gewichte sich für die Funktionsabschnitte eignen, wurden drei Kombinationen ausprobiert. Diese sind zusammen mit den dabei entstandenen Ergebnissen in Tabelle 7.4 aufgeführt. Es wurde normalisierte Zeit verwendet, da alle Pfade aufgrund der Nutzung eines Videos als Stimulus die selben zeitlichen Ausmaße haben und auf diese Weise die Festlegung der Funktionsabschnitte einfacher zu handhaben ist.

7.3.2 Auswertung und Ergebnisse

Der Versuch bestand aus vier Durchläufen, einmal ohne Gewichtung der Metrik-Parameter und dreimal mit verschiedenen Funktionsparametern. Um die Auswirkung der AOI-abhängigen Parameter auf das Ergebnis des Vergleiches zu untersuchen, soll der Vergleich in diesem Versuch einmal mit

AOIs	1	2	3	4	5	6	7	8	9	10	11	12	13	ε
1	0	2	2	2	2	2	2	2	3	3	3	3	2	4
2	2	0	1	1	2	2	2	2	3	3	3	3	2	4
3	2	1	0	1	2	2	2	2	3	3	3	3	2	4
4	2	1	1	0	2	2	2	2	3	3	3	3	2	4
5	2	2	2	2	0	1	2	2	3	3	3	3	2	4
6	2	2	2	2	1	0	2	2	3	3	3	3	2	4
7	2	2	2	2	2	2	0	1	3	3	3	3	2	4
8	2	2	2	2	2	2	1	0	3	3	3	3	2	4
9	3	3	3	3	3	3	3	3	0	1	1	1	3	4
10	3	3	3	3	3	3	3	3	1	0	1	1	3	4
11	3	3	3	3	3	3	3	3	1	1	0	1	3	4
12	3	3	3	3	3	3	3	3	1	1	1	0	3	4
13	2	2	2	2	2	2	2	2	3	3	3	3	0	4
ε	4	4	4	4	4	4	4	4	4	4	4	4	4	4

AOIs	1	2	3	4	5	6	7	8	9	10	11	12	13
1	3	1	1	1	1	1	1	1	0	0	0	0	1
2	1	3	2	2	1	1	1	1	0	0	0	0	1
3	1	2	3	2	1	1	1	1	0	0	0	0	1
4	1	2	2	3	1	1	1	1	0	0	0	0	1
5	1	1	1	1	3	2	1	1	0	0	0	0	1
6	1	1	1	1	2	3	1	1	0	0	0	0	1
7	1	1	1	1	1	1	3	2	0	0	0	0	1
8	1	1	1	1	1	1	2	3	0	0	0	0	1
9	0	0	0	0	0	0	0	0	3	2	2	2	0
10	0	0	0	0	0	0	0	0	2	3	2	2	0
11	0	0	0	0	0	0	0	0	2	2	3	2	0
12	0	0	0	0	0	0	0	0	2	2	2	3	0
13	1	1	1	1	1	1	1	1	0	0	0	0	3

Abbildung 7.3: AOI-abhängige Parameter der Metriken im dritten Versuch. Oben sind die Kosten für die Levenshtein-Distanz abgebildet, unten die Punktzahlen für den Needleman-Wunsch-Algorithmus. Die Werte ergeben sich aus der Anzahl der Gruppen in denen beide AOIs enthalten sind. Je höher diese ist, desto höher sind die Punktzahlen und desto niedriger die Kosten, was auch in dieser Graphik durch Farben verdeutlicht wurde. Bei der Levenshtein-Distanz sind zusätzlich die Kosten für Einfügen und Löschen in der letzten Zeile beziehungsweise Spalte zu sehen, für die der Wert 4 gewählt wurde.

Standardparametern und einmal mit nach Semantik bestimmten Parametern durchgeführt werden. Nach dem Vergleich werden durch Clustering zwei Gruppen gebildet, da es in der Studie zwei verschiedene Aufgaben für die Probanden gab. Es erweisen sich diejenigen Konfigurationen als geeignet, bei denen die Scanpaths möglichst korrekt nach Gruppen bezüglich des Zielobjekts der Probanden zugeordnet werden konnten.

Angesichts eines Erwartungswertes von 50 Prozent beim zufälligen Raten der Zuordnung fallen die Ergebnisse ernüchternd aus. Dies kann ein Hinweis sein, dass die auf Typ und Farbe der AOIs basierenden Metrikparameter nicht optimal gewählt wurden.

Ein Vergleich der Gewichtungsfunktionen zeigt, dass eine Betonung bestimmter zeitlicher Abschnitte in den Scanpaths durchaus einen positiven Einfluss haben kann. Dafür ist die Wahl einer passenden Funktion eine wichtige Voraussetzung. Bei der ersten Funktion $f_{0,1}$ wurden Teile der Scanpaths beim Vergleich ignoriert, was womöglich zu einem Verlust von wichtigen Ähnlichkeiten und daher

Kompression	Metrik	keine Gewichtung	a=0, b=1	a=1, b=2	a=1, b=4
deaktiviert	LCS	52	52	64	60
	D.-Levenshtein	52	52	56	56
	D.-Levenshtein (AOI-abh.)	52	52	60	56
	N.-Wunsch	60	56	60	52
	N.-Wunsch (AOI-abh.)	60	56	56	64
aktiviert	LCS	60	60	60	52
	D.-Levenshtein	60	52	60	52
	D.-Levenshtein (AOI-abh.)	-	-	-	-
	N.-Wunsch	56	56	64	52
	N.-Wunsch (AOI-abh.)	-	-	-	-

Tabelle 7.4: Ergebnisse des dritten Versuchs. In dieser Tabelle ist der prozentuale Anteil der korrekt zugeordneten Scanpaths pro Metrik und Test vermerkt, einmal für deaktivierte und einmal für aktivierte Kompression. Bei jedem Test kam eine andere Gewichtungsfunktion zum Einsatz. Bei Verwendung von Kompression ergaben sich im Schnitt höhere Werte, obwohl AOI-abhängige Parameter bei komprimierten Scanpaths nicht genutzt werden können.

zu einem schlechteren Ergebnis führte. Die zweite Funktion $f_{1,2}$ ermöglichte die besten Resultate, bei ihr wurden die als interessant festgelegten Abschnitte doppelt so hoch gewichtet wie der Rest. Eine vierfache Gewichtung bei $f_{1,4}$ verschlechterte den Ausgang des Tests. Abgesehen vom AOI-abhängigen Needleman-Wunsch-Algorithmus und der Standard-Levenshtein-Distanz wurden die Quoten dadurch schlechter im Vergleich zur zweiten Funktion.

Eine aktivierte Kompression konnte in manchen Fällen das Ergebnis verbessern, bei der Verwendung der Gewichtungsfunktion $f_{1,4}$ gelang es jedoch in keinem Fall. Das liegt womöglich an der Verkürzung der Scanpaths, die bei der Kompression nicht für jeden Pfad gleich ausfällt und eine Gewichtung der Abschnitte ungenau werden lässt.

8 Zusammenfassung, Fazit und Ausblick

In diesem letzten Kapitel werden zunächst die Ergebnisse der Evaluation sowie die daraus gefolgerten Ergebnisse zusammengefasst. Anschließen werden Möglichkeiten für weitere Arbeiten aufgezeigt.

Zusammenfassung

In den Kapiteln zu den Grundlagen (Kapitel 2 und den existierende Arbeiten (Kapitel 4) dieser Arbeit wurden verschiedene Möglichkeiten zur Vorverarbeitung von Scanpaths sowie mehrere Vergleichsmetriken analysiert. Darauf aufbauend wurde dann ein Verfahren zum Vergleich der Scanpaths mit einer Reihe von optionalen Vorverarbeitungsschritten und einer Auswahl von Metriken erarbeitet und in einem Prototypen umgesetzt.

Die Vorverarbeitungsschritte beginnen dabei mit dem bei ScanMatch (siehe Abschnitt 4.1) verwendeten Temporal Binning, welches erwartungsgemäß einen positiven Einfluss hatte. Eine Filterung von kurzen AOI-Aufenthalten konnte in keinem der Versuche die erhoffte Wirkung zeigen, kann allerdings bei leicht verrauschten Daten oder bei einem auf lange Aufenthalte begrenzten Interesse eingesetzt werden. Die Zusammenfassung von AOI-Aufenthalten zu einem Zeichen im Scanpath vereinfachte die Daten und erschwerte dadurch den Vergleich, in manchen Fällen kann ohne diesen Schritt jedoch die für die Berechnung benötigte Zeit problematisch werden. Eine Kompression von Scanpaths kann den Vergleich trotz der dadurch nicht mehr verwendbaren AOI-Informationen verbessern. Dies wurde im dritten Versuch der Evaluation festgestellt. Außerdem wird durch die Verkürzung der Scanpaths der Rechenaufwand beim Vergleich verringert.

Der Algorithmus von Needleman und Wunsch wurde von ScanMatch übernommen, zusammen mit dem Ansatz zur Erstellung der Ersetzungsmatrix basierend auf räumlichen oder farblichen Ähnlichkeiten zwischen den AOIs. Er wies eine hohe Leistungsfähigkeit auf, vor allem unter Einbeziehung von AOI-Informationen. Ein Nachteil, vor allem bei langen Scanpaths ist die hohe Laufzeit des Algorithmus. Eine Übertragung der AOI-abhängigen Parameter auf die Levenshtein-Distanz wurde vorgenommen, erbrachte jedoch nicht die erhoffte Verbesserung der Leistungsfähigkeit dieser Metrik. Bereits bei ScanMatch wurde außerdem gezeigt, dass der Needleman-Wunsch-Algorithmus der Standard-Levenshtein-Distanz überlegen ist. Die Longest Common Subsequence konnte als Scanpath-Vergleichsmetrik allenfalls bei schwach verrauschten Daten oder klar erkennbaren Ähnlichkeiten ein gutes Ergebnis liefern. In diesen Fällen kann sie jedoch durch eine einfache Verwendbarkeit und hohe Geschwindigkeit im Vergleich zu den anderen Metriken überzeugen. Da das Ausmaß der Verrauschung im Voraus kaum abzuschätzen ist, empfiehlt sich die Verwendung dieser Metrik nur in Ausnahmefällen.

Eine zeitabhängige Gewichtung der Metrikparameter kann die Ergebnisse des Vergleichs verbessern, sofern eine geeignete Funktion gewählt wird. Von den getesteten abschnittsweise definierten Funktionen erwies sich vor allem eine Gewichtung der Abschnitte mit geringer und hoher Relevanz im Verhältnis Eins zu Zwei als geeignet. Werden Abschnitte durch ein Gewicht von Null ignoriert, verschlechtert sich das Ergebnis.

Fazit

Zum aktuellen Zeitpunkt kann eine klare Empfehlung für den Algorithmus von Needleman und Wunsch unter Verwendung AOI-abhängiger Kosten gegeben werden. Trotz einer hohen Laufzeit überzeugt die Qualität der Ergebnisse im Vergleich zu denen der anderen beiden getesteten Metriken.

Eine Vorverarbeitung der Scanpaths durch Temporal Binning verbessert die Erkennung von zeitlichen Ähnlichkeiten zwischen den Pfaden. Auf die anderen Vorverarbeitungsschritte sollte nur dann vertraut werden, wenn diese im Rahmen der Studie unnötige Daten entfernen oder eine Vereinfachung der Scanpaths zur Beschleunigung der Analyse notwendig erscheint. Eine Kompression kann die Ergebnisqualität erhöhen, falls keine oder nur schwache AOI-Ähnlichkeiten als Metrikparameter vorhanden sind.

Ausblick

Es können in zukünftigen Arbeiten noch andere Kompressionsalgorithmen getestet werden, da diese womöglich ein besseres Verhalten bezüglich der Ersetzung ähnlicher Muster in den Scanpaths aufweisen können. Zusätzlich kann eine Unterstützung von unscharfen Übereinstimmungen für einen groben Vergleich der Muster dienen.

Für viele in dieser Arbeit verwendete Ansätze sind Erweiterungen möglich. Die Parameter der Metriken werden im bisherigen Konzept nach euklidischen Abständen oder Beziehungen der AOIs ermittelt. Dabei werden jedoch alle AOIs gleich behandelt, ungeachtet ihrer Größe, Form, Position oder gar der ihnen von den Probanden entgegengebrachten Aufmerksamkeit. Diese Faktoren können allerdings einen Einfluss auf die entstehenden Scanpaths haben, etwa aufgrund von Tendenzen, eher in die Mitte des Stimulus zu sehen. Je größer eine AOI ist, desto wahrscheinlicher ist es für eine Fixation, sich in dieser AOI zu befinden. Um solche Tendenzen auszugleichen, können die Metrikparameter durch weitere AOI-bezogene Daten gewichtet werden.

Für die Wahl der Gewichtungsfunktion besteht eine Vielzahl von Möglichkeiten. In dieser Arbeit wurde lediglich eine abschnittsweise definierte Funktion verwendet, die je nach Relevanz eines Abschnitts einen niedrigen oder hohen Wert ausgibt. Hier können noch weitere Ansätze ausprobiert werden, wie etwa stetige Funktionen oder eine Erhöhung beziehungsweise Verringerung der Gewichte mit zunehmender Zeit.

Die Festlegung der Parameter nach AOI-Beziehungen kann ausgeweitet werden. Der in dieser Arbeit entwickelte Prototyp betrachtet nur die Mitgliedschaft einer AOI in einer beliebigen Gruppe. Eine AOI kann jedoch in mehreren Gruppen enthalten sein, die jeweils verschiedene Ähnlichkeiten implizieren.

Auch wurde lediglich die direkte Verwandtschaft zwischen Eltern- und Kind-AOI betrachtet. Hier könnte eine transitive Betrachtung auch noch die Kinder der Kind-AOIs betreffen.

Denkbar ist auch eine Kombination mehrerer Metriken, beispielsweise mit einem Vergleich von Saliency-Maps, wie es von Le Meur und Baccino beschrieben wurde [22].

Als zusätzliche Visualisierungen könnten Zeitleisten wie in eSeeTrack (siehe Abschnitt 4.4) oder eine Parallel-Scanpath-Visualisierung (Abschnitt 4.5) in den Prototypen integriert werden.

Literaturverzeichnis

[1] S. F. Altschul and B. W. Erickson. Optimal sequence alignment using affine gap costs. *Bulletin of mathematical biology*, 48(5-6):603–616, 1986. (Zitiert auf Seite 27)

[2] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped blast and psi-blast: a new generation of protein database search programs. *Nucleic acids research*, 25(17):3389–3402, 1997. (Zitiert auf Seite 27)

[3] E. D. Cornell, H. G. Macdougall, J. Predebon, I. S. CURTHOYS, et al. Errors of binocular fixation are common in normal subjects during natural conditions. *Optometry & Vision Science*, 80(11):764–771, 2003. (Zitiert auf Seite 15)

[4] F. Cristino, S. Mathôt, J. Theeuwes, and I. D. Gilchrist. Scanmatch: A novel method for comparing fixation sequences. *Behavior research methods*, 42(3):692–700, 2010. (Zitiert auf den Seiten 26, 35, 36, 37, 38, 52, 56, 63, 72 und 74)

[5] F. J. Damerau. A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176, 1964. (Zitiert auf Seite 25)

[6] L. Dempere-Marco, X.-P. Hu, S. M. Ellis, D. M. Hansell, and G.-Z. Yang. Analysis of visual search patterns with emd metric in normalized anatomical space. *Medical Imaging, IEEE Transactions on*, 25(8):1011–1021, 2006. (Zitiert auf Seite 21)

[7] R. Dewhurst, M. Nyström, H. Jarodzka, T. Foulsham, R. Johansson, and K. Holmqvist. It depends on how you look at it: Scanpath comparison in multiple dimensions with multimatch, a vector-based approach. *Behavior research methods*, 44(4):1079–1100, 2012. (Zitiert auf den Seiten 18, 20, 21, 22, 23, 39, 42 und 63)

[8] E. W. Dijkstra. A note on two problems in connexion with graphs. *Numerische mathematik*, 1(1):269–271, 1959. (Zitiert auf Seite 39)

[9] A. T. Duchowski. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers*, 34(4):455–470, 2002. (Zitiert auf Seite 14)

[10] J. H. Goldberg and J. I. Helfman. Scanpath clustering and aggregation. In *Proceedings of the 2010 symposium on eye-tracking research & applications*, pages 227–234. ACM, 2010. (Zitiert auf den Seiten 16 und 18)

[11] J. H. Goldberg and J. I. Helfman. Scanpath clustering and aggregation. In *Proceedings of the 2010 symposium on eye-tracking research & applications*, pages 227–234. ACM, 2010. (Zitiert auf Seite 54)

[12] W. H. Gomaa and A. A. Fahmy. A survey of text similarity approaches. *International Journal of Computer Applications*, 68(13):13–18, 2013. (Zitiert auf Seite 24)

[13] R. W. Hamming. Error detecting and error correcting codes. *Bell System technical journal*, 29(2):147–160, 1950. (Zitiert auf Seite 24)

[14] D. Herr. Neue visualisierungsbasierte analysetechniken für eye-tracking-daten. 2013. (Zitiert auf Seite 67)

[15] K. Holmqvist, M. Nyström, R. Andersson, R. Dewhurst, H. Jarodzka, and J. Van de Weijer. *Eye tracking: A comprehensive guide to methods and measures*. Oxford University Press, 2011. (Zitiert auf den Seiten 12, 13, 14, 16, 19, 21, 22, 25, 40 und 59)

[16] E. B. Inc. Encyclopædia britannica 2006 ultimate reference suite dvd, 2008. (Zitiert auf Seite 12)

[17] H. Jarodzka, K. Holmqvist, and M. Nyström. A vector-based, multidimensional scanpath similarity measure. In *Proceedings of the 2010 symposium on eye-tracking research & applications*, pages 211–218. ACM, 2010. (Zitiert auf den Seiten 21, 39 und 40)

[18] M. A. Just and P. A. Carpenter. A theory of reading: from eye fixations to comprehension. *Psychological review*, 87(4):329, 1980. (Zitiert auf Seite 14)

[19] J. B. Kruskal and M. Wish. *Multidimensional scaling*, volume 11. Sage, 1978. (Zitiert auf Seite 32)

[20] K. Kurzhals, C. F. Bopp, J. Bässler, F. Ebinger, and D. Weiskopf. Benchmark data for evaluating visualization and analysis techniques for eye tracking for video stimuli. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pages 54–60. ACM, 2014. (Zitiert auf den Seiten 64 und 89)

[21] K. Kurzhals, F. Heimerl, and D. Weiskopf. Iseecube: Visual analysis of gaze data for video. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 43–50. ACM, 2014. (Zitiert auf Seite 57)

[22] O. Le Meur and T. Baccino. Methods for comparing scanpaths and saliency maps: strengths and weaknesses. *Behavior research methods*, 45(1):251–266, 2013. (Zitiert auf den Seiten 21, 22, 23 und 95)

[23] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710, 1966. (Zitiert auf Seite 24)

[24] S. P. Lloyd. Least squares quantization in pcm. *Information Theory, IEEE Transactions on*, 28(2):129–137, 1982. (Zitiert auf Seite 29)

[25] P. J. Lynch. Lateral orbit nerves. `http://commons.wikimedia.org/wiki/File:Lateral_orbit_nerves_chngd.jpg`, 2006. (Zitiert auf Seite 13)

[26] S. Mannan, K. Ruddock, and D. Wooding. Automatic control of saccadic eye movements made in visual inspection of briefly presented 2-d images. *Spatial vision*, 9(3):363–386, 1995. (Zitiert auf Seite 22)

[27] S. Mannan, K. Ruddock, and D. Wooding. Fixation sequences made during visual examination of briefly presented 2d images. *Spatial Vision*, 11(2):157–178, 1997. (Zitiert auf Seite 22)

[28] S. K. Mannan, K. H. Ruddock, and D. S. Wooding. The relationship between the locations of spatial features and those of fixations made during visual examination of briefly presented images. *Spatial Vision*, 10(3):165–188, 1996. (Zitiert auf Seite 22)

[29] C. D. Manning, P. Raghavan, H. Schütze, et al. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008. (Zitiert auf Seite 30)

[30] S. B. Needleman and C. D. Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of molecular biology*, 48(3):443–453, 1970. (Zitiert auf den Seiten 25, 27 und 36)

[31] D. Noton and L. Stark. Scanpaths in eye movements during pattern perception. *Science, New Series*, 171(3968):308–311, 1971. (Zitiert auf Seite 16)

[32] H. Nyquist. Certain topics in telegraph transmission theory. *American Institute of Electrical Engineers, Transactions of the*, 47(2):617–644, 1928. (Zitiert auf Seite 35)

[33] T. Okuda, E. Tanaka, and T. Kasai. A method for the correction of garbled words based on the levenshtein metric. *Computers, IEEE Transactions on*, 100(2):172–178, 1976. (Zitiert auf Seite 24)

[34] M. Pohl, M. Schmitt, and S. Diehl. Comparing the readability of graph layouts using eyetracking and task-oriented analysis. In *Computational Aesthetics*, pages 49–56, 2009. (Zitiert auf Seite 21)

[35] M. Raschke, X. Chen, and T. Ertl. Parallel scan-path visualization. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 165–168. ACM, 2012. (Zitiert auf den Seiten 45 und 46)

[36] M. Raschke, D. Herr, T. Blascheck, T. Ertl, M. Burch, S. Willmann, and M. Schrauf. A visual approach for scan path comparison. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, pages 135–142. ACM, 2014. (Zitiert auf den Seiten 46 und 47)

[37] P. Riordan-Eva, E. T. Cunningham, D. Vaughan, and T. Asbury. *Vaughan & Asbury's general ophthalmology. (18th ed. ed.).* McGraw-Hill Medical, 2011. (Zitiert auf Seite 11)

[38] D. Sankoff. Matching sequences under deletion/insertion constraints. *Proceedings of the National Academy of Sciences*, 69(1):4–6, 1972. (Zitiert auf Seite 28)

[39] B. Shneiderman and M. Wattenberg. Ordered treemap layouts. In *infovis*, page 73. IEEE, 2001. (Zitiert auf Seite 32)

[40] T. F. Smith and M. S. Waterman. Identification of common molecular subsequences. *Journal of molecular biology*, 147(1):195–197, 1981. (Zitiert auf Seite 27)

[41] Tobii. Tobii t60xl eyetracker user manual. http://www.tobiipro.com/product-listing/tobii-pro-t60xl/, 2011. (Zitiert auf Seite 15)

[42] H. Y. Tsang, M. Tory, and C. Swindells. eseetrack&# 8212; visualizing sequential fixation patterns. *Visualization and Computer Graphics, IEEE Transactions on*, 16(6):953–962, 2010. (Zitiert auf den Seiten 44 und 45)

[43] R. A. Wagner and R. Lowrance. An extension of the string-to-string correction problem. *Journal of the ACM (JACM)*, 22(2):177–183, 1975. (Zitiert auf Seite 25)

[44] D. S. Walther Grauman. *CompactLehrbuch der gesamten Anatomie. Band 4: Sinnessysteme, Haut, ZNS, Periphere Leitungsbahnen.* Schattauer, Stuttgart u. a., 2005. (Zitiert auf Seite 11)

[45] T. A. Welch. A technique for high-performance data compression. *Computer*, 6(17):8–19, 1984. (Zitiert auf Seite 71)

[46] J. M. West, A. R. Haake, E. P. Rozanski, and K. S. Karn. eyepatterns: software for identifying patterns and similarities across fixation sequences. In *Proceedings of the 2006 symposium on Eye tracking research & applications*, pages 149–154. ACM, 2006. (Zitiert auf den Seiten 32, 43 und 44)

[47] Wikimedia Commons. Schematic diagram of the human eye in english. http://commons.wikimedia.org/wiki/File:Schematic_diagram_of_the_human_eye_en.svg, 2007. (Zitiert auf Seite 12)

[48] Wikipedia. Longest common subsequence problem. http://en.wikipedia.org/wiki/Longest_common_subsequence_problem, Mai 2015. (Zitiert auf den Seiten 28 und 29)

[49] Wikipedia. Smith-waterman algorithm. http://en.wikipedia.org/wiki/Smith-Waterman_algorithm, Mai 2015. (Zitiert auf Seite 28)

[50] D. S. Wooding. Fixation maps: quantifying eye-movement traces. In *Proceedings of the 2002 symposium on Eye tracking research & applications*, pages 31–36. ACM, 2002. (Zitiert auf Seite 21)

[51] J. Ziv and A. Lempel. Compression of individual sequences via variable-rate coding. *Information Theory, IEEE Transactions on*, 24(5):530–536, 1978. (Zitiert auf Seite 71)

[52] D. Zuehlke. *Nutzergerechte Entwicklung von Mensch-Maschine-Systemen: Useware-Engineering für technische Systeme.* Springer-Verlag Berlin Heidelberg, 2012. (Zitiert auf Seite 11)

Alle URLs wurden zuletzt am 20. 10. 2015 geprüft.

Erklärung

Ich versichere, diese Arbeit selbstständig verfasst zu haben. Ich habe keine anderen als die angegebenen Quellen benutzt und alle wörtlich oder sinngemäß aus anderen Werken übernommene Aussagen als solche gekennzeichnet. Weder diese Arbeit noch wesentliche Teile daraus waren bisher Gegenstand eines anderen Prüfungsverfahrens. Ich habe diese Arbeit bisher weder teilweise noch vollständig veröffentlicht. Das elektronische Exemplar stimmt mit allen eingereichten Exemplaren überein.

I hereby declare that the work presented in this thesis is entirely my own. I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

Ort, Datum, Unterschrift