

Adaptive Vibrationsmuster durch Evolutionäre Algorithmen

Bachelorarbeit

von

Thomas Rzepka

an der Fakultät für Informatik

Verantwortlicher Betreuer: Prof. Dr. Michael Beigl

Betreuernder Mitarbeiter: Erik Pescara, M.Sc.

Bearbeitungszeit: 14.01.2018 – 28.05.2018

Erklärung

Hiermit erkläre ich, dass ich die vorliegende Bachelorarbeit selbstständig verfasst und keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht und weiterhin die Richtlinien des KIT zur Sicherung guter wissenschaftlicher Praxis beachtet habe.

Karlsruhe, den 28.05.2018

Inhaltsverzeichnis

1 Einleitung	1
1.1 Zielsetzung der Arbeit	1
1.2 Gliederung der Arbeit	1
2 Grundlagen	3
2.1 Taktile Geräte	3
2.2 Evolutionärer Algorithmus	3
2.2.1 Evolution in der Biologie	4
2.2.2 Selektion	4
2.2.3 Variation	4
2.2.4 Gendrift	4
2.2.5 Allgemeiner Vorgang eines Evolutionären Algorithmus	4
2.2.6 Darstellung	6
2.3 Verwandte Arbeiten	6
2.3.1 Taptic Engine	8
2.3.2 iPhone	8
2.3.3 Martian Smartwatch	9
2.3.4 Andere Hersteller	9
2.3.5 Design of a Wearable Tactile Displays	9
3 Analyse	11
3.1 Anforderungen	11
3.2 Existierende Lösungsansätze	13
4 Entwurf	15
4.1 Ausführung des Programms	15
4.1.1 Signal	15
4.2 Evolutionärer Algorithmus	16
4.2.1 Muster	20
5 Implementierung	23
5.1 Informationen zur Person	23
5.2 Signal	24
5.3 Bestimmung der Grenzen durch Bewertung durch den Nutzer	24
5.4 Ausführen des Algorithmus	25
5.5 Muster Erkennung	26
6 Evaluierung	31
6.1 Studiendesign	31
6.2 Analyse	33
6.2.1 Angaben der Probanden	33
6.2.2 Initialisierung der Grenzen	33
6.2.3 Auswertung der Iterationen des Algorithmus	35

6.2.4 Muster	42
7 Zusammenfassung und Ausblick	47
8 Anhang	49
Literaturverzeichnis	53

1. Einleitung

1.1 Zielsetzung der Arbeit

Jedes Jahr werden neue Smartphones veröffentlicht und vorgegaukelt das alles neu entwickelt wurde. Dabei ändern sich nur die inneren Komponenten durch einen neuen Prozessor und kleinen Softwareupdates sowie eine minimale Verbesserung an der Kamera. Es wird immer davon gesprochen, dass man dem Benutzer mit dem neuen Smartphone ein besseres Erlebnis bieten möchte.

Durch diese Abhandlung widmet sich die Frage, wie weit kann ein Smartphone individuell personalisiert werden. Unter den Standardeinstellungen kann man zwar ein Hintergrundbild, einzelne Klingeltöne, Schriftgrößen und die Platzierungen der Apps anpassen. Aber das ist alles nur für die visuellen sowie audiovisuellen Wahrnehmungen. Wie soll man unterschiedliche Informationen wahrnehmen, wenn das Smartphone auf dem Tisch oder in der Hosentasche ist und das Smartphone auf stumm gestellt ist?

Aus diesem Grund verfolge man das Ziel, personalisierte Vibrationssignale mittels dem vorgegebenen Wearable zu erstellen. Hierbei werden drei verschiedene Vibrationssignale für einen Nutzer so angepasst, dass die Werte speziell für den Benutzer bestimmt werden. Zur Bestimmung der Vibrationssignale wird ein Evolutionärer Algorithmus verwendet. Nachdem diese Handlungsvorschrift die passenden Werte gefunden hat, wird überprüft, wie gut die Benutzer die personalisierten Vibrationssignale im Vergleich zu vorgegebenen Werten erkennen können. Es hat sich dabei die Hypothese gebildet, dass die personalisierten Vibrationen besser erkannt werden als die vorgegebenen Vibrationen.

1.2 Gliederung der Arbeit

Im Verlauf dieser Bachelorarbeit erläutere ich erst allgemein, was die einzelnen Bestandteile des Evolutionären Algorithmus sind und wie ich dies an mein Problem angepasst und implementiert habe. Die Erkenntnisse sind im letzten Abschnitt zusammengefasst. Im Ausblick wird beschrieben, was man damit bewirkt und erkannt hat und wie man diese Erkenntnisse auf die zukünftigen Projekte anwenden kann. Des weiteren werde ich auf die Umsetzungen der heutigen Hersteller von personalisierten Vibrationsmuster eingehen.

2. Grundlagen

2.1 Taktile Geräte

Ein Taktiles Gerät ist ein Gerät, dass Informationen an einen Menschen durch die Wahrnehmung der Haut mitteilt.[\[GOS01\]](#)

Taktile Geräte werden heutzutage öfter verwendet, als man es eigentlich wahrnimmt. Ein einfaches Beispiel ist das Smartphone. Eine Person trägt sein eigenes Smartphone in der Hosentasche. Bei einer eingehenden Nachricht, muss der Benutzer mitgeteilt werden, dass eine Nachricht empfangen wurde. Normalerweise geschieht das beim Abspielen des Klingeltons. Falls man beschäftigt ist und nicht durch ein lautes Klingeln gestört werden will, stellt man den Ton ab. Um dennoch den Nutzer darauf Aufmerksam zu machen, dass eine Nachricht eingetroffen ist, wird die Vibration des Smartphones verwendet.

Ein weiteres Beispiel in der Taktile Displays benutzt werden ist für Blinde Menschen, um Wissen zu übermitteln [\[PB03\]](#). Für Blinde gibt es spezielle Karten, die Konturen für die Umrisse der Länder darstellen, solche Karten sind selten vertreten. Um ein solches Ungleichgewicht zwischen den blinden und normal sehenden Schülern zu mindern, hat man sich ein System entwickelt, dass audiovisuelle und taktile Eindrücke vereint. Dabei hat man sich für eines räumlich akustisches Soundsystem entschieden, um den Schüler anhand Geräuschen aus verschiedenen Richtungen ein Gefühl zu geben, an welchen Orten man sich aktuell befindet und was um einen herum passiert. Es wurden nicht nur Tier- und Verkehrgeräusche verwendet, sondern auch den Namen der Region, in der man sich aktuell befindet, wurde ausgesprochen. Als Taktiles Device hat man sich an schon existierenden Mäusen und Controllern bedient, die Force-Feedback besaßen.

Die verwendeten Karten waren nicht so detailliert wie in der Realität. Im Vergleich zu der normalen Karten hat man hier weniger Informationen, die man übertragen muss und die Details der Ländergrenzen konnte man leichter darstellen.

Damit ein Schüler eine Karte erkunden konnte, bewegt er sich mithilfe eines Eingabegeräts über die Karte und drückt eine Taste auf der Tastatur um Informationen über das Gebiet abzurufen, diese wurden über Audios übermittelt. Das Taktile Device wurde verwendet um Grenzübergänge zu mittels Vibrationen zu signalisieren.

2.2 Evolutionärer Algorithmus

Unter dem Begriff der Evolutionären Algorithmen (EA) versteht man eine Ansammlung von Techniken und Methoden, die für Optimierungsprobleme eine Lösung findet, die das

Problem näherungsweise löst. [Wei15]

2.2.1 Evolution in der Biologie

Im 19. Jahrhundert hat sich Darwin mit den Gedanken über die Evolution von Lebewesen über mehrere Generationen gemacht. Dabei kam er auf das Prinzip „Survival of the fittest“. Mit dem Prinzip hat Charles Darwin die Entstehung neuer Arten beschrieben. Es haben die Arten überlebt, die sich am besten für die Umgebung angepasst haben. Die Arten, die sich an die neue Situation nicht angepasst hatten, sind nach ein paar Generationen zur Minderheit geworden. [Hün07]

In der Biologie ist jeder lebende Organismus ein **Individuum**. Jedes Individuum besitzt Erbinformationen in der Form von Chromosomen. Die Erbinformationen werden auch **Gene** oder **DNA** genannt. Eine Gruppe von Individuen wird als **Population** bezeichnet.

Bei dem EA hat man sich das Verhalten in der Biologie angeschaut, wie die Arten über Generationen hinweg überleben und versucht dieses zu adaptieren. [Fli]

2.2.2 Selektion

Eine **Selektion** ist eine Auswahl von Individuen einer Population. Durch Kombination der Gene der ausgewählten Individuen werden neue Individuen für die nächste Generation erzeugt. Diese Kombination wird **Rekombination** genannt. Es gibt verschiedene Selektionsstrategien. Man versucht die Individuen zu finden, die eine best mögliche Lösung für ein Problem liefern. [Wei15, Fli]

2.2.3 Variation

Für eine Population sollte man zu Beginn eine große **Variation** von Individuen mit unterschiedlichen Genen besitzen. Anhand eines Beispiels nehme man an, dass man mithilfe einer Evolution eine neue Art von Süßigkeiten entwickeln möchte. Dabei sollten die Form und Farbe neu bestimmt werden. Wenn man jetzt eine Start-Population von Individuen habe, die alle die gleichen Gene besitzen würden, so würde man anhand zweier selektierten Individuen keine Änderung der nächsten Generation erkennen, da die Nachkommen auch alle die gleichen Gene besitzen, falls keine Mutation auftritt. Damit dieses Verhalten nicht auftritt, versucht man zu Beginn eine große Variation an Individuen zu erzeugen und diese als Anfangs-Population für einen Evolutionären Algorithmus zu nutzen. Mittels Rekombination und Mutation wird dabei ein neues Individuum für die nächste Generation erzeugt. Mithilfe der Rekombination werden verschiedene Gene bei der Fortpflanzung der nächsten Generation vermischt und somit neue Gene erzeugt. Die Mutation ist eine Veränderung der Gene eines Individuums, die meistens durch Umwelteinflüsse ausgelöst werden. [Fli]

2.2.4 Gendrift

Unter **Gendrift** versteht man eine zufällige Veränderung der Genhäufigkeit in einer Population, innerhalb einer Evolution. Ein Gendrift tritt bei kleineren Populationen häufiger auf. [bro5]

2.2.5 Allgemeiner Vorgang eines Evolutionären Algorithmus

Der EA besitzt im Allgemeinen immer die gleichen Komponenten. Diese Komponenten werden im Folgenden erläutert. Dieses Wissen stammt aus den folgenden Quellen: [Shi12, Fli, Wei15]

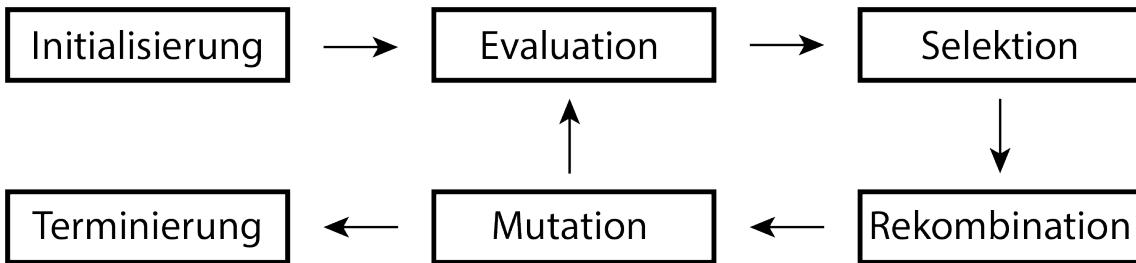


Abbildung 2.1: Ablauf des Evolutionären Algorithmus

Initialisierung

In der Initialisierung erzeugt man eine Population von Individuen, die eine zahlreiche Variationen von Genen besitzen. Für die Anfangs-Population nimmt man in der Praxis zufällige Individuen oder auch die besten bekannten Lösungskandidaten.

Bewertung der Individuen

Bevor man eine Evolution für eine Population erzeugen kann, benötigt man Attribute, anhand derer man die nächste Generation berechnet. Mit einer sogenannten **Fitnessfunktion** berechnet man mittels der Attribute einen **Fitnesswert**. Dieser ist entscheidend für die Selektion.

Selektion

Die Selektion wählt zwei Individuen als Eltern aus. Es werden die Individuen bevorzugt, die einen hohen Fitnesswert aufweisen. Diese ermittelten Eltern werden mittels der Rekombination und Mutation weiter verarbeitet.

Rekombination

Die selektierten Eltern werden in diesem Schritt einen Nachfahren für die nächste Generation erzeugen. Dabei werden die jeweiligen Gene der Eltern kombiniert und an die nächste Generation vererbt. Die Kombinationsmöglichkeiten hängen davon ab, wie die Gene repräsentiert sind. Die neuen Individuen werden in die Population für die nächste Generation hinzugefügt.

Mutation

Nach der Rekombination besteht eine Chance, dass die Gene des Nachkommens mutieren können.

Terminierung

Man führt für eine Population die Selektion, Rekombination und Mutation so oft aus, bis man die gleiche Anzahl an Individuen für die nächste Generation erzeugt hat. Der Vorgang der Evaluierung, Selektion, Rekombination und Mutation muss für jede neue Generation durchgeführt werden. Die neu erzeugte Population wird als Eingabe für die nächste Generation verwendet. Dabei wird der Algorithmus so lange ausgeführt, bis eine hinreichende Abbruchbedingung erreicht worden ist. Je nach Definition wird entweder nach einer festen Anzahl an Generationen oder erst nachdem es keine signifikanten Änderungen mehr gibt, der EA terminiert. Das dadurch erzeugte Ergebnis ist eine näherungsweise Lösung für das Problem.

1	2	3	4	5	6	7	8
1	0	1	1	0	0	1	0

Abbildung 2.2: Darstellung einer binären Repräsentation der Länge 8

Tabelle 2.1: Beispiel für eine diskrete und intermediäre Rekombination

Elternteil A	0,3	0,9	0,1	0,5
Elternteil B	0,6	0,2	0,8	0,5
Nachkomme direkt	0,3	0,2	0,1	0,5
Nachkomme intermediär	0,45	0,55	0,45	0,5

2.2.6 Darstellung

Es gibt verschiedene Darstellungsmöglichkeiten von Genen, die meist verwendeten Darstellungen werden im Folgenden erläutert. Das wissen, wurde aus den Quellen [Fli, Shi12, Dil17] entnommen.

Lineare Repräsentation

Eine **Lineare Repräsentation** besteht aus einer Folge $a_0a_1a_2\dots a_{n-1}$ von Symbolen einer festen Länge n , die aus einer definierten Symbolmenge V entnommen wurde, dabei ist $a_i \in V$.

Die binäre Repräsentation Abbildung 2.2 besitzt die Symbolmenge $V = \{0,1\}$ und ist die meist verwendete *lineare Repräsentation*.

Falls man nur aus einem Elternteil ein Nachkomme erzeugen soll, wird dabei die Rekombination übersprungen und es wird mit der Mutation fortgefahrene. Es gibt zwei Möglichkeiten die Nachfolger der zwei Eltern zu erzeugen. Bei der *diskreten Rekombination* wählt man die Werte eines Gens für das Nachkommen aus, die besser geeignet sind. Am Beispiel Tabelle 2.1 wurden nur die kleineren Werte für das Nachkommen bevorzugt. Die andere Möglichkeit ist die *intermediäre Rekombination*. Aus den Attributen der Elternteile wird der Durchschnitt gebildet und an das Nachkommen vererbt.

Beim *crossover* erhält man aus zwei Eltern zwei Nachkommen. Das Single-point crossover überschneidet die beiden Eltern an einer Stelle und die resultierenden Eltern werden als Nachkommen übernommen. Beim Two-point crossover hat man zwei Überschneidungen der Eltern, sowie man bei dem uniform crossover jedes zweite Element mit dem Partner getauscht hat.

Baumstrukturen

Für **baumartige Strukturen**, wird die Genetische Programmierung verwendet. Man verwendet die baumartigen Strukturen um Formeln oder Programme zu repräsentieren. Die Selektion, Rekombination und Mutation muss auf der Baumstruktur ausgeführt werden. Als Beispiel für eine Baumstruktur repräsentiert die Abbildung 2.4 den mathematischen Ausdruck $4 - 23 * 2$.

2.3 Verwandte Arbeiten

Da heutzutage beinahe jedes Gerät ein Vibrationsmotor verbaut hat, sei es das Handy, die Smartwatches oder Fitnessarmbänder (u.v.m.), werde ich im folgenden auf einige aktuelle

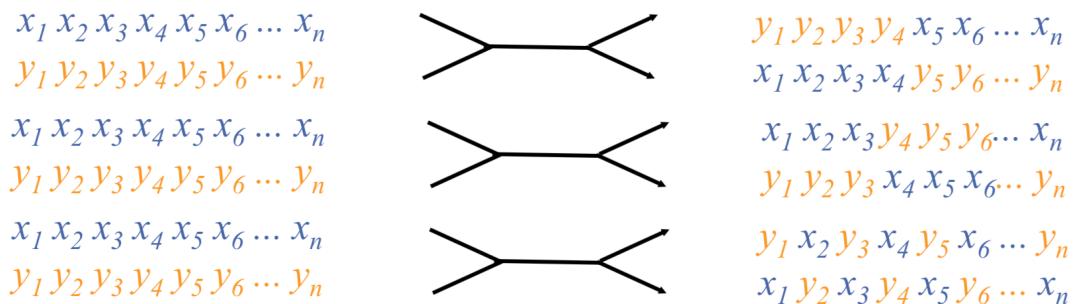


Abbildung 2.3: Darstellung eines Single-point, Two-point und uniform crossovers (von oben nach unten)

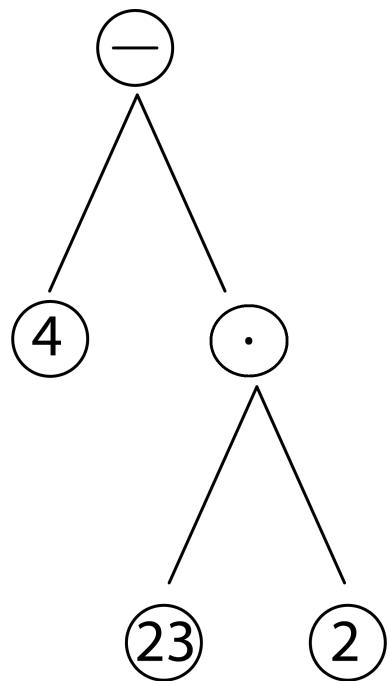


Abbildung 2.4: Baumstruktur des mathematischen Ausdrucks $4 - 23 * 2$

Technologien und deren Umsetzung der personalisierten Vibrationsmuster zu sprechen kommen.

2.3.1 Taptic Engine

Die Taptic Engine ist ein von der Firma Apple selbst entwickeltes Vibrationsmotor, dass heutzutage in nahezu allen Apple Produkten verbaut ist. Das erste Gerät, was die Taptic Engine bekommen hat, war die Apple Watch. Der Name **Taptic** bildet sich aus dem Wörtern „Taktil“ und „Haptisch“. Trotz der neu Erfindung einer mechanischen Rückmeldung, bietet Apple keine Personalisierung, wie lange eine Rückmeldung für die Apple Watch erfolgen soll. Die Einstellungsmöglichkeiten an der Apple Watch ist lediglich die Stärke der Vibration. Diese ist in 3 Stärkestufen unterteilt. Meiner Ansicht nach kann man daher nicht wirklich von einer personalisierten Vibration sprechen.



Abbildung 2.5: Einstellungsmöglichkeit auf der Apple Watch (links) und in der Martian App (rechts)

2.3.2 iPhone

Der Hersteller Apple hat auch bei dem iPhone eine Möglichkeit geboten, eigene Vibrationsmuster zu erstellen, jedoch mit Einschränkungen. Wenn man in die jeweilige Einstellung der iPhones gelangt, erscheint das folgende Bild Abbildung 2.6. Beim Drücken auf das Display wird an der Stelle eine Vibration erzeugt. Man hat 10 Sekunden um ein eigenes Muster zu erzeugen, indem man wiederholt auf den Bildschirm drückt. An der Stelle, an der man den Bildschirm berührt hat, erscheint visuell um der Position ein Kreis. Die erzeugten Vibrationen werden in einer Leiste visuell angezeigt. Man kann sich beliebig viele Vibrationsmuster speichern, die bis zu 10 Sekunden lang sind. [FSDS16]

Die Einschränkung, die man hier erwähnen muss ist, dass man die Vibrationsmuster nur für Systeminterne Funktionen benutzen kann. Dies bedeutet, dass man die Funktionen für Klingeltöne, Nachrichtentöne, Erinnerungshinweise, Kalenderhinweise (o. ä.) hinzufügen kann. Für eine andere Anwendung, die nicht im Betriebssystem integriert sind, ist das nicht möglich. Somit können Benachrichtigungen von anderen Entwicklern keine eigenen Vibrationsmuster erhalten. Trägt man das iPhone in der Hosentasche und es wird eine Benachrichtigung einer Applikation empfangen, die nicht im System integriert ist, kann man anhand der Vibrationen des iPhones nicht unterscheiden, welche Application dies gewesen ist.

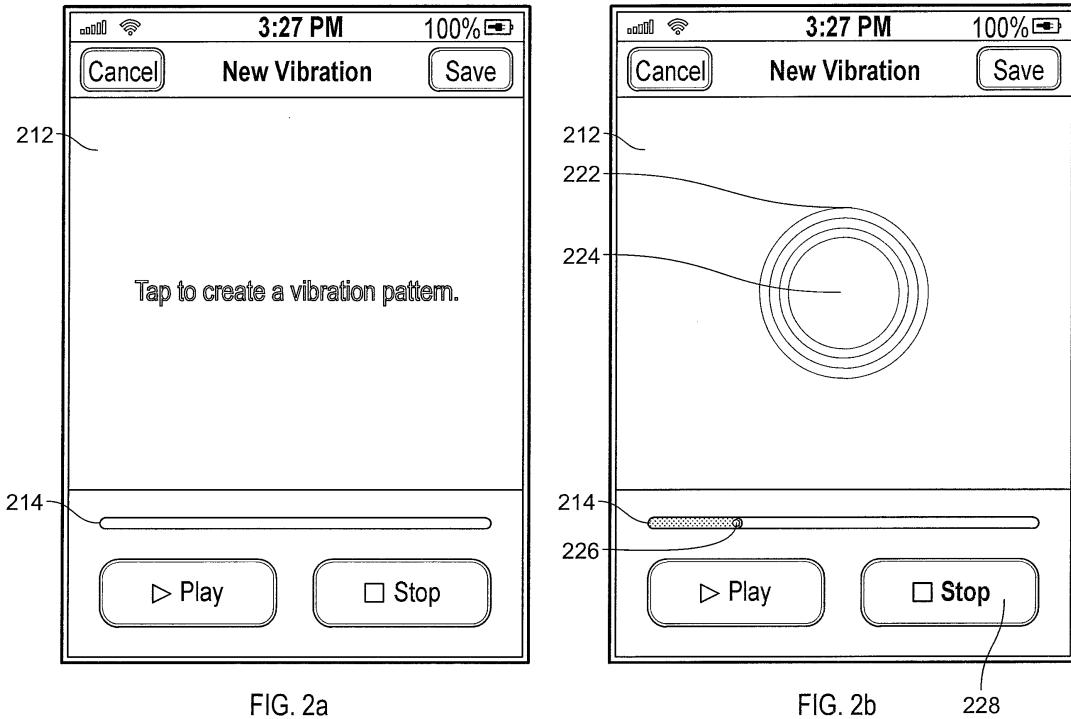


Abbildung 2.6: Erstellung eigener Vibrationsmuster auf dem iPhone

2.3.3 Martian Smartwatch

Das Gerät, das es nach meiner Recherche am besten gelöst hat, ist eine Smartwatch von einem kleinen Startup namens Martian. Das Startup hat eine Uhr hergestellt, mit der man mittels einer App auf dem Smartphone die Vibrationsmuster selbst anpassen kann. Die App unterstützt eine große Anzahl an Applikationen, von anderen Herstellern, die Benachrichtigungen senden. Ein Vibrationsmuster für die Uhr kann man aus mit zu 4 Signalen auf der Uhr darstellen. Die Signale sind als Lang, Kurz und Pause festgelegt. Somit kann man mittels der Vibration der Uhr herausfinden, welche App gerade eine Benachrichtigung auf mein Handy gesendet hat. Die Länge und Stärke eines Signals sind vom Hersteller festgelegt und kann nicht geändert werden.

2.3.4 Andere Hersteller

Bei sehr vielen Herstellern ist es aktuell noch gar nicht möglich eigene Vibrationsmuster zu erstellen. Bei Android Geräten ist es aktuell so, dass man aus einer Menge von wenigen vordefinierten Vibrationsmustern sich nur einen auswählen kann. Einige Entwickler haben dieses Problem erkannt und eigene Applikationen entwickelt, die im Store veröffentlicht sind.

2.3.5 Design of a Wearable Tactile Displays

In dem Paper von [GOS01] handelt es um Taktile Displays; was man bei der Erschaffung von Taktilem Display beachten soll und was für Verwendungszwecke es noch gibt. Das Paper ist schon mehr als 15 Jahre alt und weist dennoch auf Informationen hin, die heutzutage noch Relevant sind. Die Inhalte des Papers werden im Folgenden beschrieben. Wie oben bereits erwähnt wurde, sind Taktile Displays Geräte, die dazu benutzt werden

um Informationen durch Körperkontakt des Menschen über Haptisches Feedback zu übermitteln. Diese bilden keinen Konflikt mit den audio-/visuellen Einflüssen. Die Taktiles Displays sind eine Unterstützung der Darstellung der Informationen. Beispielsweise kann man Blinden oder Tauben Informationen mittels Taktiles Displays vermitteln, die Sie nicht wahrnehmen können. Dabei werden haptische / sensorische assistive Geräte benutzt, um die Informationen aus der realen Umgebung in taktile Simulationen umzuwandeln. Ein wichtiger Aspekt in diesem Paper ist es gewesen, wie man ein solches Taktiles Display entwirft. Außerdem gibt es Aspekte, die man beim Entwurf beachten sollte:

- leise, leicht und klein sein
- Reduzierung des Stromverbrauchs
- die Taktoren sollten durch die Kleidung spürbar sein
- so eng wie möglich am Körper liegt

Seit langem haben Handys bereits Vibrationsmotoren, um den Nutzer darauf Aufmerksam zu machen, falls eine Nachricht erhalten wurde. Die Vibration des Handys war eine Metapher dazu, dass eine andere Person einem an der Schulter rütteln würde. [GOS01] Um es nicht nur Theoretisch zu erklären hat man ein Taktiles Display als Weste entworfen, bei dem man die Vibrationsmotoren aus Nokia Handys verwendet hat. Mittels der Weste sollte eine Person vom Punkt A zu Punkt B navigiert werden. Die übermittelten Informationen zur Navigation sind vorwärts, zurück, rechts, links, beschleunigen und verlangsamten gewesen.

3. Analyse

3.1 Anforderungen

Die Aufgabenstellung bestand darin, herauszufinden, ob ein personalisiertes Vibrationsmuster besser als ein vorgegebenes Vibrationsmuster erkannt wird. Die Grundvoraussetzung sollte ein System sein, dass mit dem Wearable kommuniziert und Daten übertragen kann, sodass das Wearable Vibrationen abspielt. Darüber hinaus soll eine Studie erstellt werden, in der man herausfindet, ob es einen signifikanten Unterschied zwischen generischen und genetischen Vibrationen gibt. Die generischen Vibrationen sind fest vorgegeben, wobei die genetischen Vibrationen, anhand der Bewertungen des Probanden angepasst werden sollen. Dabei habe man ein Evolutionären Algorithmus verwendet, um die genetischen Werte für einen Probanden zu bestimmen.

Zunächst musste man sich im Vorfeld ein paar Gedanken über die Repräsentation eines Signals, die Dekodierung und die Übertragung machen. Im Folgenden werden diese Entscheidungsfindungen beschrieben.

Wearable

Damit man Vibrationen überhaupt darstellen kann, hätte man sich ein eigenes Taktils Device entwerfen müssen. In meinem Fall war dies nicht nötig, da das Wearable vom

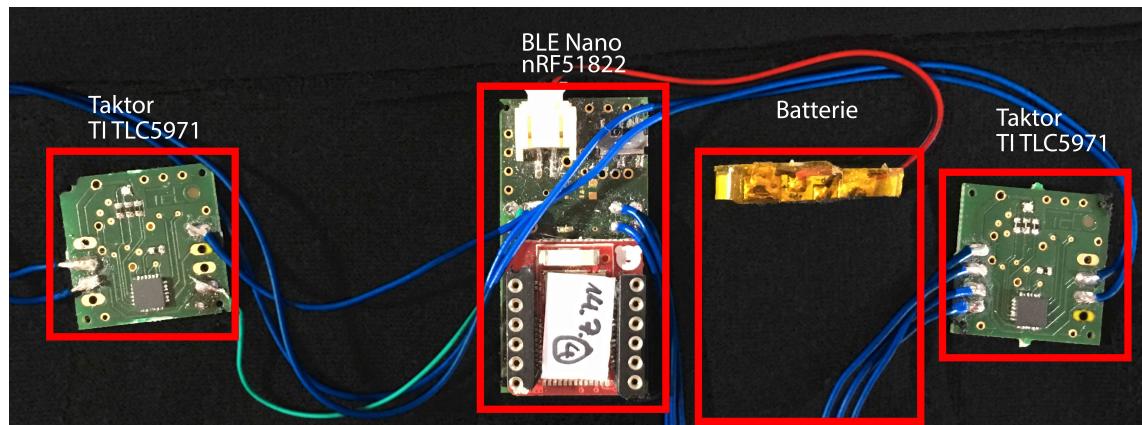


Abbildung 3.1: Armband geöffnet. Die Komponenten sind zwei Traktoren, eine Batterie und ein Mikroprozessor

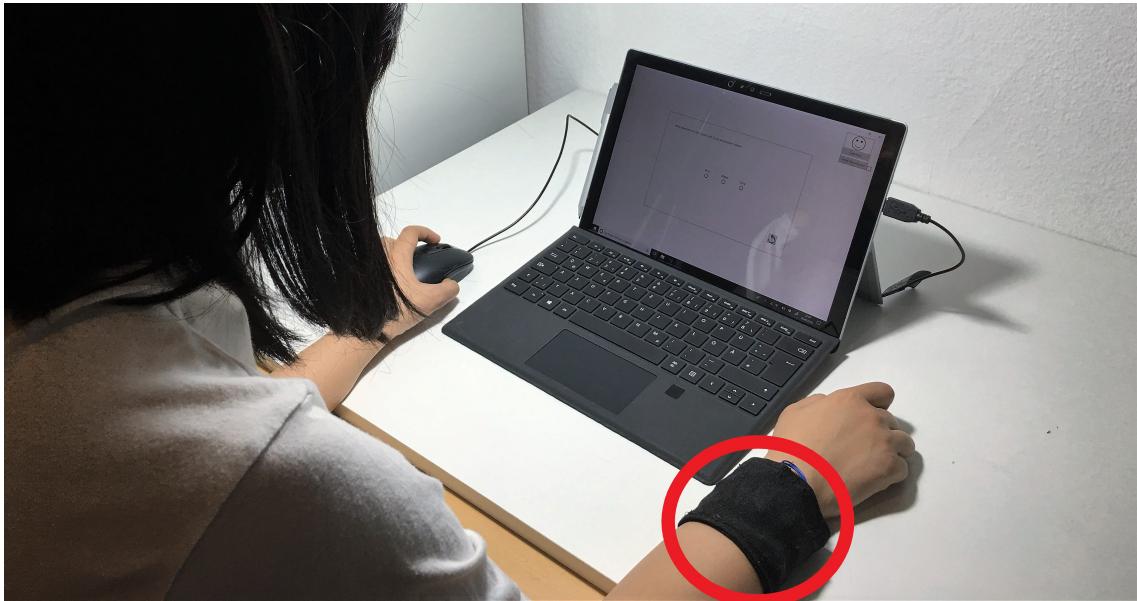


Abbildung 3.2: Proband während der Durchführung der Studie.

TECO vorgegeben war. Zu den Komponenten des Wearables gehören zwei Taktoren (TI TLC5971), einen Mikroprozessor (nRF51822 / BLE Nano) und eine Batterie.

Die Kommunikation des Armbands konnte man über eine serielle Schnittstelle, sowie über Bluetooth Low Energie (LE) realisieren. Bei der seriellen Schnittstelle ist ein USB-Kabel nötig gewesen um den PC mit dem Wearable anzusprechen, dabei wäre die Mobilität verloren gegangen. Daher habe man die verfügbare BLE Schnittstelle benutzt um zwischen dem PC und dem Wearable zu kommunizieren.

Bluetooth LE überträgt Nachrichten nur in einer maximalen Größe von 20 Bytes. Diese werden jeweils als zwei Bytes Blöcke übertragen.

Anhand der Begrenzung der Datenübertragung hat man sich eine geeignete Dekodierung des Signals überlegen müssen.

Die Programmiersprache um die Kommunikation zwischen dem PC und dem Wearable aufzubauen, konnte frei gewählt werden. Dafür habe man zuvor verschiedene Programmiersprachen untersucht. Java hat man nach fehlenden Informationen zum Kommunikationsaufbau ausgeschlossen. Schließlich hat man sich für C# entschieden, da hier ausführliche Dokumentationen vorhanden waren. Außerdem hat man sich für diese Programmiersprache wegen ihrer funktionsfähigen Kommunikation und der einfachen Grafischen Benutzeroberfläche entschieden. Die Grafische Benutzeroberfläche hat man für die Studie verwendet.

Dabei musste herausfinden, was für Einstellungsmöglichkeiten das Wearable besitzt. Diese waren die Länge und die Stärke eines Signals.

Signale

Ein Signal repräsentiert eine Vibration, die auf dem Wearable abgespielt wird. Es hat sich die Frage gestellt, wie personalisiert man denn jetzt Signale. Hier hat man das Signal in drei verschiedene Längen definiert, **Kurz**, **Mittel** und **Lang**.

Dabei hatte man als Vorgabe, dass man einen Evolutionären Algorithmus (EA) benutzen sollte. Diesen EA hat man verwendet, um für die Signaltypen einen personalisierten Wert zu bestimmen.

Man hat sich ein Vorgang überlegen müssen, um für den EA eine Startpopulation zu erzeugen; die Individuen der Population zu bewerten; mittels einer Fitnessfunktion den Fitnesswert für jedes Individuum zu berechnen; eine geeignete Selektion, Reproduktion und Mutation, sowie die Terminierungsbedingung zu bestimmen.

Eine geeignete Signalrepräsentation musste definiert werden, um Signale vom PC zum Wearable über BLE zu senden und abspielen zu lassen. Das bedeutet sowohl am PC als auch am Wearable selbst dies definiert werden.

Studie

Es sollte eine Grafische Benutzeroberfläche erstellt werden, damit der Proband die Studie komplett über das Programm ausführen kann, indem er die Instruktionen des Programms folgt.

Es hat sich durch das Definieren der Signale und dem EA ergeben, dass die Studie in drei Schritten unterteilt wird. In dem ersten Schritt muss man Grenzen für die Erzeugung einer Startpopulation für einen EA bestimmen. Der zweite Schritt ist es, einen genetischen Wert für die eigentliche Studie zu bestimmen, dies passiert durch Ausführen mehrerer Generationen des EA. Als letzten Schritt definiert man eine Folge von generischen und genetischen Signalen, die man anschließend vom Benutzer bewerten lässt.

Alle Daten sollten überprüft werden und es sollte herausgefunden werden, ob es am signifikante Unterschiede gibt.

3.2 Existierende Lösungsansätze

Rüttelflug

Aus dem Paper von [PBB16] wurde ein Wearable für das Paragliding entworfen. Beim Entwurf des Wearables hat man sich an den vorhin genannten Anforderungen von [GOS01] gehalten. Das Wearable besitzt einen Traktor oberhalb und einen unterhalb Handgelenks und wurde sowohl in einer Laborstudie als auch in einer Feldstudie getestet. Für die Laborstudie wurden Vibrationssignale in der Länge von 100, 200, 400, und 800ms abgespielt. Jeder Proband hat dieselbe Reihenfolge von Aufgaben ausgeführt bekommen. Eine der Aufgaben der Probanden war den Namen und die Länge des Signals zuzuordnen. Aus der Laborstudie hat sich ergeben, dass die Probanden die Signallänge 800ms mit einer Genauigkeit von 97,6% sowie die beiden Singnalslänge 200 und 400ms zu 100% richtig erkannt haben.

Die generischen Werte für die Erkennung der Muster hat man aus dem Paper übernommen.

4. Entwurf

4.1 Ausführung des Programms

Zu Beginn hat man sich über das gesamte Problem einen Überblick schaffen müssen. Dabei hat man sich alle einzelnen Bestandteile im Detail angeguckt, um herauszufinden welche Komponenten man wie zusammensetzt. Es haben sich drei / vier große Komponenten ergeben, die man erledigen musste.

Zuerst habe man sich mit dem Wearable auseinandergesetzt, um zu bestimmen, was man alles mit dem Wearable umsetzen kann. Man fand dabei heraus, dass man zwei Parameter hatte, die man einstellen konnte. Diese waren die Zeit, wie lang das Wearable vibriert und die Stärke, wie stark das Armband vibriert. Um ein Signal zu übertragen hatte man eine Beschränkung von 20 Bytes, die man maximal in einer Nachricht mittels Bluetooth übertragen kann.

Als nächste Komponente des Problems habe man sich eine geeignete Repräsentation eines Signals überlegen müssen. Um nicht außer Bedacht zu lassen ist diese die wichtigste Komponente des Problems, denn es hängen alle anderen Komponenten davon ab, wie diese Daten repräsentiert werden.

Daher stellte sich die Frage, wie sieht ein solches Signal aus?

4.1.1 Signal

Anhand der zwei Parameter, die Länge und die Stärke des Wearables, hat man diese als Attribute eines Signals definiert.

Technisch gesehen hätte man für die Länge eines Signals von 0 ms (0x0000) bis 65535 ms (0xFFFF) nutzen können. Man hat sich jedoch anhand vorheriger Studien [PBB16] orientiert und ein Minimum von 100 ms und ein Maximum von 1000 ms als Grenzen für die Länge übernommen.

Bei der Stärke eines Signals hat man sich auch hier am Wearable orientiert. Die Repräsentation der Stärke musste man aus technischen Gründen ab einem Wert von 32767 (0x7FFF) definieren. Dies war nämlich der Wert, ab dem beide Traktoren angefangen haben spürbar, als Vibrationen wahrgenommen zu werden. Vor dem genannten Wert ist zu wenig Strom übertragen worden, sodass kein Traktor funktionierte. Daher hat man sich zwischen den Grenzen von (0x7FFF) und (0xFFFF) für die Stärke festgelegt.

Tabelle 4.1: Bewertung eines Probanden die Ideal ist. Dabei ist die *Länge* die Signallänge in ms und *Erkannt* ist das erkannte Signal durch den Benutzer

Länge	100	200	300	400	500	600	700	800	900	1000
Erkannt	Kurz	Kurz	Kurz	Mittel	Mittel	Mittel	Lang	Lang	Lang	Lang

Anhand der Hypothese der Bachelorarbeit wolle man wissen, wie gut sich personalisierte Vibrationen im Vergleich zu generischen Vibrationen verbessern. Dabei musste man sich überlegen, wie man personalisierte Vibrationen mittels dem EA für einen Probanden bestimmen will. Außerdem sollte man herausfinden, was für Signale noch gut voneinander unterscheidbar sind, die die Grenzen von 100ms bis 1000ms besitzen. Man habe sich auf drei Typen von Signalen festgelegt: **Kurz**, **Mittel** und **Lang**. Dadurch wolle man wissen, wie ein Signal wahrgenommen wird und welchen Typen man dem Signal zuordnen würde.

Die jeweiligen Typen definieren innerhalb der Grenzen zwischen 100 ms und 1000 ms ein Intervall, die sich nicht überschneiden. Abgesehen davon ist **kurz**, das zahlenwertig kleinste Intervall, gefolgt von **mittel**, **lang** ist das zahlenwertig größte Intervall. Allerdings wurde darauf geachtet, dass die Grenzen nicht aufeinander liegen, sondern einen Abstand zwischen den Grenzen existiert. Als Beispiel habe man für Kurz die Intervallgrenzen 100ms und 300ms, für Mittel habe man dann die Intervallgrenzen von 400ms bis 600ms und für Lang habe man die Intervallgrenzen von 700ms bis 1000ms.

Für die Stärke der Vibration war man an den Darstellbarkeit des Wearables gebunden. Zwischen die Grenzen von (0x7FFF) und (0xFFFF) hat man sich fünf voneinander unterscheidbare Stufen definiert.

4.2 Evolutionärer Algorithmus

Grenzen Initialisieren

Als Eingabe für den Algorithmus benötigt man eine Anfangspopulation von Individuen. Die Individuen sind in diesem Fall Signale. Nicht jede Person empfindet ein vorgegebenes kurzes Signal gleich wie eine andere Person, somit musste man zuvor den Benutzer befragen, was er als Kurz, Mittel und Lang empfindet. Zuerst habe man alle Signale abgespielt, damit der Proband wusste, was Ihn erwartet. Nachdem alle Signale abgespielt wurden, wurde jedes Signal einzeln abgespielt und der Proband hatte die Aufgabe das abgespielte Signal der Kategorie Kurz, Mittel oder Lang zuzuordnen. Jedem Probanden wurde insgesamt 10 Signale mit der gleichen Vibrationsstärke abgespielt, die Signallänge ist dabei gleich verteilt. Dabei wurden alle 10 Signale in einer zufälligen Reihenfolge abgespielt.

Nach der Eingabe des Probanden erhält man beispielsweise folgende Bewertung [Tabelle 4.1] erhalten. Diese Eingabe ist ideal, da alle Signaltypen direkt hintereinander vorliegen, so würde man hier einfach die Grenzen aus der Tabelle entnehmen können. Diese belaufen sich für Kurz zwischen 100 und 300 ms, für Mittel zwischen 400ms und 600ms und für Lang zwischen 700 und 1000ms.

Falls die Eingabe jedoch nicht so Ideal sein sollte, wie im Beispiel [Tabelle 4.2], mussten ein paar Vorkehrungen getroffen werden. Um einige Sonderfälle auszuschließen, hat man überprüft, ob das Signal mit der Länge von 100ms ein Kurzes Signal ist und das Signal mit der Länge von 1000ms ein Langes Signal ist, sowie man annimmt, dass mindestens jeder Signaltyp mindesten zweimal ausgewählt wurde. Sollte dies nicht der Fall sein, so würde man den Benutzer dazu bitten, die zehn Signale erneut zu bewerten.

Man beginnt damit die neuen Intervallgrenzen zu bestimmen. Diese wurden anhand des Beispiels [Tabelle 4.3] exemplarisch bestimmt, dabei ist jede Spalte eine Iteration. Dabei

Tabelle 4.2: Bewertung eines Probanden die nicht Ideal ist. Dabei ist die *Länge* die Signallänge in ms und *Erkannt* ist das erkannte Signal durch den Benutzer

Länge	100	200	300	400	500	600	700	800	900	1000
Erkannt	Kurz	Kurz	Mittel	Mittel	Lang	Mittel	Lang	Lang	Lang	Lang

Tabelle 4.3: Exemplatische Bestimmung der Grenzen

Iterationen	1.	2.	3.	4.	5.	6.	7.	8.	10.
<i>KurzMin</i>	100	100	100	100	100	100	100	100	100
<i>KurzMax</i>	100	200	200	200	200	200	200	200	200
<i>MittelMin</i>	100	200	300	300	300	300	300	300	300
<i>MittelMax</i>	100	200	300	400	400	600	600	600	600
<i>LangMin</i>	100	200	300	400	500	600	700	700	700
<i>LangMax</i>	100	200	300	400	500	600	700	800	1000

bestimmt man zuerst den kleinsten Wert und setzt den für alle Grenzen gleich. Falls ein größerer Wert des für den gleichen Signaltypen existiert, erhalten alle nachfolgenden Grenzen den gleichen Wert zugewiesen. Wenn für den nächsten Signaltyp ein größerer Wert vorliegt, werden nur noch die nachfolgenden Signaltypen nach diesem Wert angepasst. Nachdem die 10 Iterationen ausgeführt sind, untersucht man, ob die Grenzen nicht identisch sind und ob die Grenzen jeweils aufsteigend sind.

$$\text{Kurz}_{\text{Min}} < \text{Kurz}_{\text{Max}} < \text{Mittel}_{\text{Min}} < \text{Mittel}_{\text{Max}} < \text{Lang}_{\text{Min}} < \text{Lang}_{\text{Max}}$$

Falls dies nicht der Fall sein sollte, dann wird der Benutzer erneut aufgefordert die 10 Signale zu bewerten.

Bei einer kleinen Abweichung von zwei nebeneinanderliegen von zwei Werten ist dies noch akzeptabel, bei größeren Abweichungen, hat man Benutzer noch einmal darum gebeten erneut zu bewerten. Im Verlauf der Studie musste man nur 16% aller Probanden ein weiteres Mal darum bitten, die Signale neu zu bewerten.

Population

Sind die Grenzen für einen Probanden bestimmt, kann darauffolgend die Startpopulation erzeugt werden. Bei der **Startpopulation** hat man für jeden Signaltypen zehn Signale. Zwei Signale der zehn Signale repräsentieren das Minimum und das Maximum des jeweiligen Signaltypen. Die restlichen acht Signale erhalten eine zufällige Länge innerhalb des Intervalls. Die Stärke eines Signals wird zufällig für jedes Signal zufällig zugewiesen. Man besitzt somit für die Startpopulation dreißig Signale, die jeweils in zehn Kurze, Mittlere und Lange Signale unterteilt sind.

An dieser Stelle muss man besonders betonen, dass man nur beim ersten Mal eine **Startpopulation** erzeugt. Für die Erzeugung der Populationen für die nächsten Generationen werden die Grenzen der Signaltypen nicht explizit hinzugefügt.

Bewertung des Algorithmus

Jedes Individuum einer Population muss vor der Ausführung des nächsten Schritts des Algorithmus bewertet werden. Um diese Bewertung zu erhalten hätte man aus einigen Alternativen wählen können wobei man hier ein weiteres Erläutern werde und wieso man dieses nicht gewählt hat.

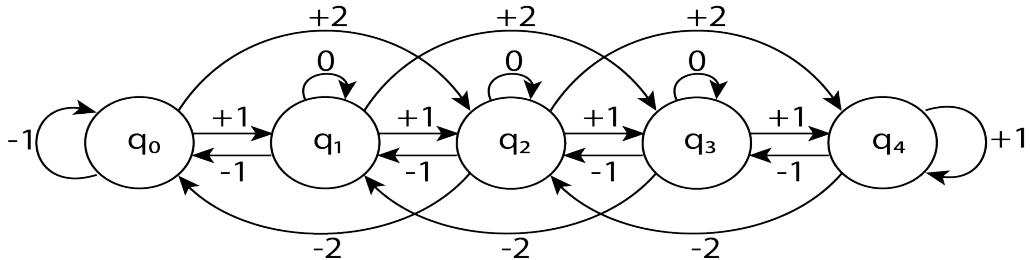


Abbildung 4.1: Zustandsdiagramm für die Stärke

Zuerst könnte man hergehen und jedes Signal aus der Population dem Probanden abspielen und fragen was für ein Signaltyp er erkannt hat, wie man es bei den ersten zehn Signalen gemacht hat, um die Grenzen der Signaltypen zu bestimmen.

Der erste Ansatz wäre man würde die Anzahl der richtigen Zuweisungen und Abweichungen zählen, um mit einer geeigneten Fitnessfunktion den Fitnesswert zu bestimmen. Ferner würde das bedeuten, dass für eine Population ein Individuum mehrmals abgespielt werden müsse, um eine Bewertung des einzelnen Individuums zu erhalten. Bei einer Annahme von fünf Bewertungen pro Individuum, um nur eine Population damit bewerten zu können, müsste der Proband eine Anzahl von 150 Signalen bewerten, damit der Algorithmus eine Generation bestimmen kann. Dies würde bedeuten, dass wenn man vier Generationen bestimmen wollen würde, man auf eine Anzahl von 600 Vibratoren kommt, die der Benutzer bewerten sollte, um einen personalisierten Wert zu erhalten.

Aufgrund der Tatsache, dass Probanden nicht so lange an einer freiwilligen Studie teilnehmen wollen, habe man sich für einen anderen Ansatz entschieden. Man muss akzeptieren, dass man über mehrere Generationen hinweg die komplette Population bewerten muss. Das heißt, dass man pro Generation 30 Signale abspielen muss. Um eine Generation zu bestimmen habe man sich Gedanken darüber gemacht, wie man so ein Signal bewerten soll. Dabei ist man zu dem Entschluss gekommen, dass man den Benutzer zuerst fragt, was für ein Signaltyp er erkannt habe. Weiterhin habe man Ihnen zwei Fragen wie in einem Fragebogen (Likert-Skala) gestellt, wie gut er das Signal erkannt hat und wie man die Stärke des Signals empfand.

Wie schon erwähnt hat jedes Individuum einer Population eine zufällige Stärke zugewiesen. Dabei entspricht die Stärke einem Zustand aus dem Zustandsdiagramm [Abbildung 4.1], das in der ersten Population zufällig bestimmt worden ist, weshalb auch kein Startpunkt zu sehen ist, da jeder Zustand der Startzustand sein kann. Anhand der Bewertung auf die Frage, wie man die Stärke empfunden hat, verändert sich der aktuelle Zustand der Stärke eines aktuellen Individuums um bis zu zwei Zustände. Dabei ist die Wichtigkeit der Antwort in der [Tabelle 4.5] beschrieben. Die Bewertung +2 bedeutet, dass ein Zustandswechsel des aktuellen Zustands um zwei Zustände passiert. Um es an einem Beispiel zu erklären, nehmen wir an, dass der Startzustand q_0 ist. Dem Probanden würde jetzt ein Signal mit der Stärke 0x7FFF abgespielt. Wenn der Benutzer jetzt auf die Frage mit **zu schwach** beantwortet hätte, wechselt der Zustand jetzt zu q_2 . Wenn man immer noch von dem zufälligen Startzustand von q_0 ausgeht und der Proband nach dem Abspielen des Signals mit einer der Antworten **ok, stark** oder **zu stark** antworten würde, würde es keine Änderung des Zustandes geben, da es keine schwächeren Stärke existiert. Die Ausgabe des Zustandsdiagramms ist die Signalstärke, die in der [Tabelle 4.4] steht. Es wird zu jedem Zustand eine andere Signalstärke abgespielt. Diese Änderung erfolgt direkt nach der Beantwortung des Signals.

Anhand der Frage, was für ein Signal erkannt wurde, ist für das Programm irrelevant gewesen. Diese Information sollte für die Bewertung der Studie dienen.

Tabelle 4.4: Belegung der Signalstärke

Signalstärke	Namen	Zustandsnamen
0x7FFF	Very Weak	q_0
0x9FFF	Weak	q_1
0xBFFF	OK	q_2
0xDFFF	Strong	q_3
0xFFFF	Very Strong	q_4

Tabelle 4.5: Bewertung der Frage, wie man die Stärke empfunden hat

Antwort	zu schwach	schwach	ok	stark	zu stark
Wertigkeit	+2	+1	0	-1	-2

Mittels der letzten Frage, wie gut ein Proband das Signal erkannt hat, hat man einen Wert hinter jeder Antwortmöglichkeit gelegt. Die Antwort auf die Frage repräsentiert den Fitnesswert.

Erzeugung der nächsten Generation

Im Anschluss nach der Bewertung erfolgt die Bestimmung der nächsten Generation. Dies erfolgt mit der Selektion, Rekombination und Mutation der Gene.

Selektion

Für die Selektion wird für jeden Signaltyp ein neuer Pool erzeugt. Für jeden Signaltypen bestimmt man das Maximum $Fitness_{Max}$ von allen Fitnesswerten. Anhand der Formel

$$\text{Häufigkeit} = \frac{\text{Fitnesswert eines Individuums}}{\text{Fitness}_{Max} \text{ des Signaltyps}} \cdot 100 \quad (4.1)$$

lässt sich die Häufigkeit bestimmen, wie oft ein Individuum in den Pool hinzugefügt wird. Als Beispiel hat man ein Maximum von 5 und einen Fitnesswert von 2, daraus ergibt sich 40; das heißt, es wird 40 mal das jeweilige Individuum in den Pool hinzufügt. Dabei wird jedes Individuum in dem Pool hinzugefügt, welchem Signaltyp es angehört.

Nach der Erzeugung der Pools wird eine Rekombination durch zwei zufällig ausgewählte Individuen ausgeführt.

Rekombination

In der Rekombination wurden die zwei zufällig ausgewählten Individuen als Eltern definiert. Die gefundenen Eltern erzeugen ein neues Individuum, dass in die Population der nächsten Generation hinzugefügt wird.

Dieser Vorgang wurde wie folgt für das Problem angepasst. Man hat die zwei Gene, die Stärke und die Länge eines Signals.

Zuerst beginnen wir mit der Länge des Signals. Es wird die Länge des ersten Elternteils und die Länge des zweiten Elternteils genommen und ein Mittelwert von beiden Werten

Tabelle 4.6: Bewertung der Frage, wie gut man das Signal erkannt hat.

Antwort	sehr schlecht	schlecht	ok	gut	sehr gut
Wertigkeit	1	2	3	4	5

gebildet. Der Mittelwert ist die Länge des neuen Individuums. Bei der Stärke wurde direkt nach der Beantwortung des Signals die Stärke angepasst. Dieser angepasste Stärke wert wird von beiden Elternteilen genommen und man bildet aus beiden ebenfalls den Mittelwert und rundet diesen ab.

Aus jedem der drei erzeugten Pools werden zehn neue Individuen für die nächste Generation erzeugt, sodass man wieder auf eine Gesamtzahl von 30 Individuen für eine Population kommt.

Mutation

Es besteht eine Chance von 5%, dass sich ein Individuum mutieren kann. Das bedeutet in diesem Fall, dass sich lediglich die Länge zufällig verändern kann.

Abbruchkriterium

Man wiederholt für jede Generation wie den Zyklus [Abbildung 2.1](#), bis man eine ausreichende Lösung erhält.

Das Abbruchkriterium ist in diesem Fall erreicht, wenn die vierte Generation erzeugt worden ist. Um hier die Stellung zu der Entscheidung von vier Generationen zu nehmen, ist es so, dass man versucht habe die Anzahl der Vibrationen für einen Probanden niedrig zu belassen. Im Vergleich zum ersten Ansatz, in der man 600 Vibrationen für vier Generationen hat, habe man, für die aktuell benutzte Lösung bei 30 Individuen pro Population, für 4 Generationen, 120 Vibrationen.

Nach dem man die letzte Population vom Algorithmus erhält, bestimmt man für jeden Signaltypen das Minimum und Maximum von der Stärke und der Länge von allen Signalen. Anhand derer man den Mittelwert bestimmt und schließlich den personalisierten Wert für den Probanden erhält.

4.2.1 Muster

Damit man schließlich die Hypothese der Bachelorarbeit beantworten kann, muss man zuvor die generische Vibrationen mit genetische Vibrationen vergleichen. Man hat sich im Vorfeld drei Typen von Mustern definiert. Der erste Typ war das dreier Muster. Dabei hat man drei Signale hintereinander erzeugt, die jeweils mit einer Pause getrennt sind. Für das dreier Muster hat man sich auf eine Anzahl von 15 Muster festgelegt. Diese 15 Muster wurden zweimal erstellt, einmal mit den personalisierten Kurz, Mittel und Lang Werten, sowie die jeweiligen personalisierten Stärke Werten, die nach der Ausführung des Evolutionären Algorithmus bestimmt wurden. Das andere mal hat man sich an dem Paper [\[PBB16\]](#) für Kurz, Mittel und Lang jeweils die Werte 200ms, 400ms und 800ms, sowie die Stärke für *Strong* entschieden. Beim vierer Muster hatte man vier Signale und drei Pausen und beim fünfer Muster hatte man fünf Signale und vier Pausen. Die vierer und fünfer Muster wurden genau wie das dreier Muster erstellt. Man hat also für jeden Muster-Typen 30 Muster die auf jeweils 15 generischen und 15 genetischen Mustern bestehen.

Man hat sich als erstes überlegt für jeden Probanden zufällige Muster zu erstellen. Die Pausen der Muster habe man im Vorfeld fest definiert. Somit haben die dreier Muster entweder 100ms oder 200ms Pausen besessen. Die vierer Muster hatten Pausen, die entweder 100ms, 200ms oder 300ms gewesen sind. Das selbe gilt auch für die fünfer Muster, nur dass hier die Pausen wie beim vierer Muster und zusätzlich noch 400ms sein konnten. Die Pausen wurden für jeden Probanden einmalig festgelegt.

Damals war der Gedankengang so, dass bei den zufällig erzeugten Mustern man vermeiden wollte, dass Muster zweimal abgespielt werden und aus dem Grund habe man sich

die festen Pausen definiert, die für jeden Probanden gleich sind. Als man sich darüber Gedanken gemacht hat, wie man am Ende die Muster / Signale vergleichen wollen würde, hat man festgestellt, dass es nicht sinnvoll sei zufällige Muster zu erzeugen, da man in der Auswertung keine Vergleiche untereinander machen könnte. Zu dem Zeitpunkt hatte man die festen Pausen schon implementiert und bei der Änderung nicht mehr im Blickwinkel gehabt. So hat man nach der Änderung für alle feste Muster mit festen Pausen, die jedoch zwischen den oben genannten Werten variieren. Dies ist eines der Fehlentscheidungen, die ich erst nach der Durchführung der Studie erkannt hatte. Im Nachhinein würde ich für jeden Muster-Typen immer gleichlange Pausen zwischen den Signalen definieren.

Während der Studie wurden zuerst alle dreier Muster, gefolgt von allen vierer Muster und schließlich allen fünfer Mustern abgespielt. Es wurde abwechselnd generische und genetische Muster abgespielt. Ein Muster wurde zufällig abgespielt und es wurde kein Muster zweimal abgespielt. Die Probanden hatten die Aufgabe die Signale aus einem Muster in erkannter Reihenfolge anzugeben.

Man hat dem Probanden immer die Möglichkeit geboten, ein Signal oder Muster erneut abzuspielen.

5. Implementierung

Das Kapitel **Implementierung** wird an dem im Entwurf besprochenen Verlauf der Studie erklärt.

Bei der Entwicklung des Programms hat man sich Gedanken über ein System gemacht, bei dem man alle Daten schon digital abgespeichert hat und nicht noch Informationen vom Probanden von Fragebögen per Hand in den PC eintragen muss. Aus diesem Grund hat man sich für eine Grafische Benutzeroberfläche, auch kurz GUI (graphical user interface) genannt, entschieden. Die Programmiersprache, die verwendet wurde, ist C# gewesen. Der Entstehungsprozess wurde auf Papier entworfen. Das Design sollte schlicht und einfach sein und den Fokus auf das Programm und nicht auf andere Komponenten des Betriebssystems ablenken. Die Papier-Skizzen sind im Anhang vorhanden. Aus diesem Grund hat man sich einen schwarzen Rahmen um den Bereich gemacht, damit der Proband nur Focus darauf lenkt. Die folgenden Bilder sind nur die wesentlichen Ausschnitte des Programms. Das Programm wurde für das Surface 4 angepasst und könnte daher bei anderen Displayauflösungen anders platziert werden. Auf die Implementierung vom BLE und dem Wearable wird hier nicht weiter eingegangen.

5.1 Informationen zur Person

Bei den Nachforschungen des Studiendesigns [BTT05], habe man sich nach mehreren Entwürfen für folgende Fragen entschieden:

- Wie alt sind Sie?
- Angabe des Geschlechts
- Empfinden Sie sich als musikalisch?
- Spielen Sie gelegentlich Computer Spiele?
- Haben Sie Erfahrungen mit Taktilem Geräten?
- Haben Sie schon einmal eine Smartwatch verwendet?

Da man für den Probanden nicht mit allen Fragen auf einmal überhäufen wollte, hat man diese Fragen auf zwei Seiten dargestellt [Abbildung 6.2]. Bei der Frage um das Geschlecht habe man ebenfalls drei Antwortmöglichkeiten geboten, ob man *männlich*, *weiblich* oder

keine Angabe. Im Hintergrund hatte man eine einfache Klasse **Person**, die die Informationen der Antworten dieser Befragung gespeichert hat. Um zu vermeiden, dass Daten verloren gehen, habe man die Daten zur Person schon nach diesem Schritt in einer externen Datei gespeichert. Für jeden Probanden hat man im Vorfeld eine anonymisierte Datei angelegt, die im Nachhinein nicht mehr auf den Probanden zurückschließen konnte.

5.2 Signal

Wie schon im Entwurf beschrieben, ist das Signal eines der wichtigsten Bestandteile des Programms.

Ein **Signal** besitzt die Attribute Länge in ms, Stärke, die den jeweiligen Zustand der Stärke speichert [Abbildung 4.1](#), den Signaltypen, sowie die Grenzen des Signaltyps und die Anzahl wie oft ein Signal erneut abgespielt wurde. Da man in den nächsten beiden Phasen dem Signal einen Signaltypen zuordnen soll, wurde dafür auch ein weiteres Attribut angelegt, sowie die Zeit, die benötigt wurde, um die Frage zu beantworten.

In der Phase, indem der Algorithmus durchgeführt wird, werden noch zwei zusätzliche Fragen gestellt, für die man in der Klasse **Signal** auch noch diese beiden Werte gespeichert hat.

5.3 Bestimmung der Grenzen durch Bewertung durch den Nutzer

Nachdem der Proband die Angaben zur Person beantwortet hat, erscheint ein Benachrichtigungsfenster, indem Informationen über den die aktuelle Phase erklärt wird. Falls der Proband zu einem Zeitpunkt Fragen haben sollte, werden diese durch den Aufseher beantwortet. Für die Bestimmung der Grenzen wurden 10 Signale erzeugt, die gleich verteilt sind. Mithilfe einer Zufallsfunktion werden diese Signale zufällig ausgewählt und in dieser Reihenfolge abgespielt. Der Proband soll für jeden der 10 Signalen einen der Signaltypen *Kurz*, *Mittel* oder *Lang* zuordnen. Dabei habe man die im Entwurf besprochenen Sonderfälle überprüft und bei erfolgreicher Bildung der Grenzen ist man in die nächste Phase übergegangen.

Wenn die Bestimmung der Grenzen nicht erfolgreich gewesen ist, wird der Benutzer gebeten die gleiche zufällige Reihenfolge erneut zu bewerten.

Sollte ein Proband den Replay-Button drücken, wird das letzte Signal erneut abgespielt.

Für die GUI [Abbildung 5.2](#) hat man sich auf das wesentliche beschränkt. Die Anweisung an den Benutzer war immer dieselbe, daher hat man nur den gleichen Text dargestellt. Als Antwortmöglichkeiten des Signals wurde für jeden Signaltyp je ein Radio-Button erstellt. In den Phasen, in denen ein Signal abgespielt wurde, wurde der Replay-Button immer an der gleichen feste Position platziert.

Der Ablauf einer Bewertung wird im Folgenden beschrieben. Nach der Bestätigung des Benachrichtigungsfenster wurden im Hintergrund die 10 Signale in zufälliger Reihenfolge ausgewählt. Darauf wurde das erste Signal über BLE an das Wearable gesendet. Das Wearable hat dieses Signal empfangen, ausgewertet und hat die Informationen als Vibration abgespielt. Dass was der Benutzer dadurch empfunden hat, wurde über den PC mittels der Maus bewertet. Nach einer Bewertung gab es keine Möglichkeit die Bewertung eines vorherigen Signals erneut anzupassen. Nachdem das Signal vom Probanden bewertet wurde, wurde der Maus-Cursor durch das Programm an eine Anfangsposition gesetzt, damit der Benutzer nicht unabsichtlich doppelt auf ein Radio-Button drücken konnte. Um dem

Probanden zu signalisieren, dass er die Checkbox gedrückt hat, hat man die Check-Box einen Augenblick visuell markiert gelassen, nachdem man den Maus-Cursor neu positioniert hat. Mit der Neupositionierung der Maus hat man erreicht, dass man immer den gleichen Weg hatte um die Checkboxen zu erreichen, aber man hatte auch nach mehreren Fragen die Maus anheben weiter nach oben legen müssen, denn man hat schon das Ende der Tischkannte erreicht. Nach einer Bewertung wurde die vorherige Auswahl entfernt und man hat das nächste Signal übertragen.

5.4 Ausführen des Algorithmus

Für den Algorithmus habe man sich mehrere Klassen erzeugt. Die Klasse **DNA** beinhaltet ein Signal und Hilfsfunktionen, die eine Rekombination und eine Mutation von zwei Signalen erzeugen. In der Klasse **Population** hat man eine Liste aus 30 Elementen, von einer Klasse **DNA** erstellt, dabei sind beinhaltet die ersten 10 DNA-Objekte Signale, die den Signaltypen *Kurz* haben. Für die nachfolgenden DNA-Objekte gilt dasselbe nur mit *Mittel* gefolgt von *Lang*. Bei der Erzeugung der Signale habe man mittels einer Zufallsfunktion sich Signale mit einer zufälligen Stärke und einer zufälligen Länge zwischen den Grenzen des jeweiligen Signaltypens erstellt. Beim ersten Aufruf der Klasse *Population* habe man die Grenzen der Signaltypen ebenfalls als ein Signal erzeugt und in der Liste hinzugefügt.

Für die Grafische Benutzeroberfläche [Abbildung 5.3](#) hat man sich für die drei folgenden Fragen entschieden:

- Was für einen Signaltypen haben Sie erkannt?
- Wie gut haben Sie das Signal erkannt?
- Wie empfanden Sie die Stärke des Signals?

Es ist bis auf die Anzahl der Fragen genau das gleiche Prozedere gewesen. Es wurde zuerst ein Erklärungsfenster dargestellt, sodass der Benutzer weiß, was in dieser Phase von Ihm erwartet wird. Dabei wird im Hintergrund wie gerade erwähnt eine Population erzeugt. Diese Population wird DNA für DNA abgearbeitet. Dabei wird für jede DNA enthaltene Signal über BLE an das Wearable gesendet und die Information in Vibration umgewandelt. Dabei hat der Benutzer die drei Fragen zu dem Signal beantworten sollen. Es musste die komplette Population bewertet werden, es wurde keine DNA erneut bewertet. Man habe dabei zufällig eine DNA aus der Population gewählt, die bewertet werden sollte.

Die erste Frage diente nur dazu um in der späteren Auswertung herauszufinden, was der Benutzer im Vergleich zum Programm erkannt hat. Die zweite Frage wurde als eigentlicher Fitnesswert gespeichert. Die Repräsentation der Frage ist in der Tabelle [Tabelle 4.6](#) zu sehen. Mit der Stärke hat man direkt nach der Beantwortung aller drei Fragen nach dem Zustandsdiagramm den Zustand gewechselt.

Die Informationen der Stärke und des Fitnesswerts habe man in der Klasse *Signal* noch mitgespeichert.

Nachdem eine Population komplett berechnet wurde, habe man den Generations-zähler um eins erhöht, die bisherigen Informationen in der externen Datei gespeichert und eine neue Population aus den Werten erzeugt.

Selektion

Für jeden Signaltypen legt man einen Pool aus DNA-Objekten an, dessen Signale den jeweiligen Signaltypen besitzen. Mithilfe der Formel (im Entwurf) berechnet man die Anzahl, wie oft ein DNA-Objekt in das jeweilige Pool hinzugefügt werden soll. Man legt genauso

viele DNA-Objekte, wie berechnet in das Pool. Im Durchschnitt besitzt ein Pool nach dem Hinzufügen der Häufigkeit aller DNA-Objekte in etwa bis zu 1000 Elemente. Die Selektion erfolgt per Zufallsprinzip, dabei wählt man zwei zufällige DNA-Objekte aus einem Pool. Diese beiden DNA-Objekte werden auch als *Eltern* bezeichnet. Die beiden Eltern erzeugen nun durch Ausführung der *Rekombination* und *Mutation* einen Nachkommen für die Population der nächsten Generation. Dabei wird es so lange wiederholt, bis man genauso viele Individuen wie aus der Population der vorherigen Generation vorhanden waren.

Rekombination

In der Rekombination verwendet man die Eltern, greift man auf die jeweiligen Signale zu und verwendet die beiden Attribute Länge und Stärke. Für die beide Attribute bildet man den Mittelwert, die ganzzahlig abgerundet werden. Diese Attribute bilden werden für die Erzeugung des neuen Individuums verwendet.

Mutation und Termination

Es existiert eine fünf Prozentige Chance, dass das neu erzeugte Individuum durch ein zufällig generiertes Signal ersetzt wird.

Nachdem die vierte Generation erzeugt wurde, wird der Algorithmus terminiert. Aus der zuletzt erzeugten Population werden für die Länge und Stärke die Grenzen der drei Signaltypen bestimmt. Die Grenzen dienen als Eingabe für die Erzeugung der personalisierten Muster.

Die gerade bestimmten Mittelwerte bilden die personalisierten Werte, die für die Musterkennung verwendet werden.

5.5 Muster Erkennung

Für die Erzeugung der Muster hat man die Grenzen mit übergeben müssen, denn in der Klasse *Muster* wird der personalisierte Wert bestimmt, dies geschieht, indem man Mittelwert der Grenzen berechnet. Im Vorfeld hat man drei Typen von Mustern erstellt. Für jeden Muster-Typen hat man 15 Muster angelegt. Für jeden Muster-Typen habe man zwei Listen angelegt, dabei wurden alle 15 Muster hineingelegt mit dem einzigen Unterschied, dass in der einen Liste die Muster mit personalisierten Werten und in der anderen Liste die Muster mit den generischen Werten vorhanden sind. Die generischen Werte für *Kurz*, *Mittel* und *Lang* sind 200ms, 400ms und 800ms gewesen, alle Signale wurden mit dem Wert *Stark* für die Stärke initialisiert. Als Nächstes wurden die Listen in zufälliger Reihenfolge geordnet. Die Initialisierung hat einige Sekunden in Anspruch genommen, da man jedoch vermeiden wollte, dass zur Laufzeit ein Fehler dadurch passieren würde. Diese Wartezeit hat man akzeptiert und dem Probanden darauf aufmerksam gemacht.

Die Muster wurden abwechselnd zwischen den personalisierten und generischen Muster an das Wearable gesendet, dabei hat man die Muster in der Reihenfolge von den kleinsten bis zum größten Muster-Typen abgespielt. Die Probanden hatten die Aufgabe die Reihenfolge der Signaltypen eines Musters anzugeben. Für die GUI Abbildung 5.4 hat man nur die Bedienelemente anzeigen lassen, die aktuell benötigt waren, man habe nämlich für jeden Signaltypen einen Button gehabt. Durch Drücken auf einen dieser Buttons ist der Signaltyp als Eingabe erfolgt und es wurden die Eingaben visuell dargestellt. Der Proband hat genau die Anzahl an Signalen einzugeben gehabt, wie auch das Muster lang war, es verschwanden die drei Signal-Eingabe-Buttons und zum Vorschein kam ein Bestätigungsbutton. Erst als dieser Button gedrückt wurde, wurde das nächste Muster abgespielt und der Button verschwand und die anderen kamen zum Vorschein. Mittels einem *Clear*-Button hat man

die komplette Eingabe entfernen können um diese bei Fehlern erneut bewerten zu können. Für jedes Muster habe man die Möglichkeit gehabt, das Muster erneut abzuspielen. Bei einem Wechsel der Muster-Typen wurde ein Signal vorher dem Probanden über ein Benachrichtigungsfenster darauf aufmerksam gemacht.

Während dem Ablauf der Erkennung der Muster wurden noch die Zeit, die benötigt wurde, um ein Muster zu bewerten gespeichert.

Angaben zur Person			
Alter	<input type="text"/>		
Geschlecht	<input type="radio"/> Männlich	<input type="radio"/> Weiblich	<input type="radio"/> N.A.
Empfinden Sie sich als musikalisch?	<input type="radio"/> Ja	<input type="radio"/> Nein	
<input type="button" value="Bestätigen"/>			
Allgemeine Fragen			
Spielen Sie gelegentlich Spiele?	<input type="radio"/> Ja	<input type="radio"/> Nein	
Haben Sie Erfahrungen mit Taktilem Geräten ?	<input type="radio"/> Ja	<input type="radio"/> Nein	
Haben Sie schon einmal eine Smart watch verwendet ?	<input type="radio"/> Ja	<input type="radio"/> Nein	
<input type="button" value="Bestätigen"/>			

Abbildung 5.1: GUI für die Befragung zu den Angaben zur Person

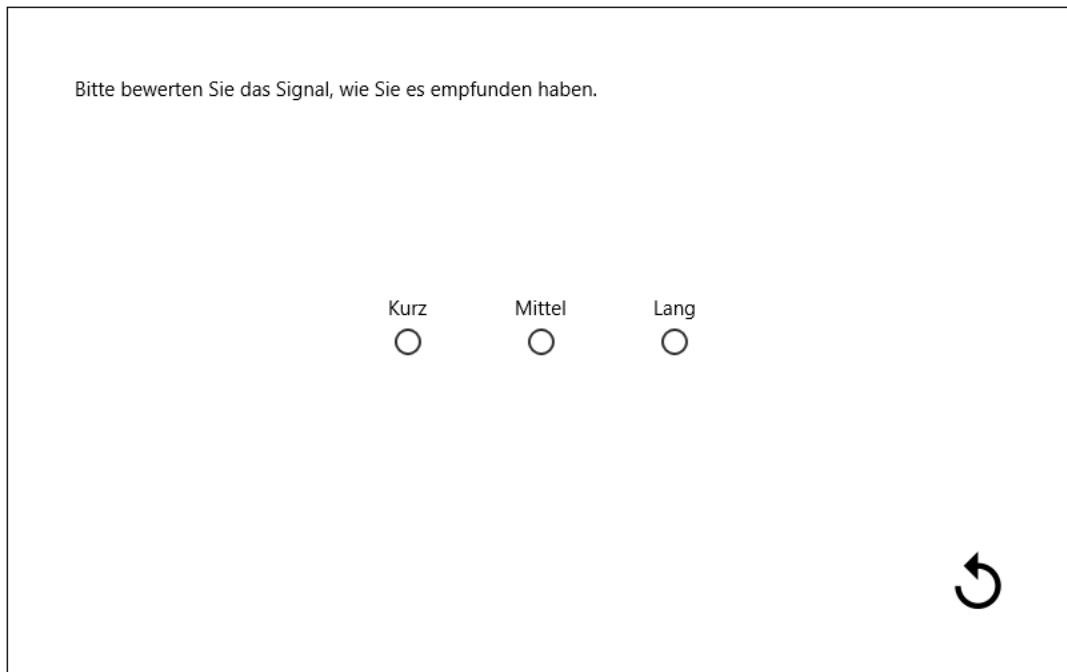


Abbildung 5.2: GUI zur Bestimmung der Grenzen

Bitte bewerten Sie das Signal.

Was für einen Signaltypen haben Sie erkannt?

kurz mittel lang

Wie gut haben Sie das Signal erkannt?

sehr schlecht sehr gut

Wie empfanden Sie die Stärke des Signals?

zu schwach zu stark

Stimmung Bewerten

Bitte bewerten Sie Ihre Stimmung

Abbildung 5.3: GUI des Algorithmus (oben) und der Bewertung der Stimmung (unten)

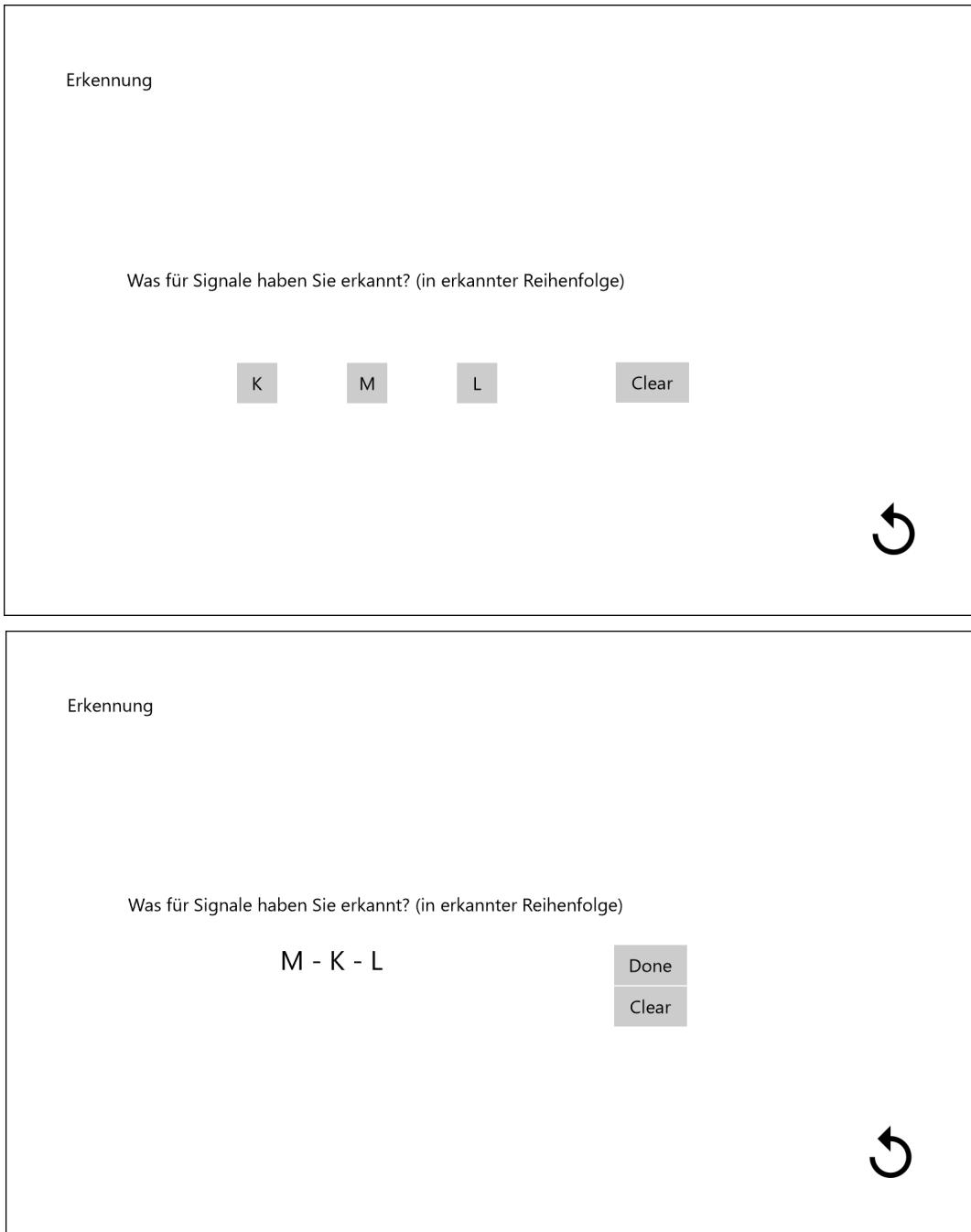


Abbildung 5.4: GUI des Musters ohne Eingabe (oben) und nach der Eingabe (unten)

6. Evaluierung

6.1 Studiendesign

Anhand dem folgenden Programmablauf [Abbildung 6.1](#) hat man sich orientiert, um die Studie zu entwerfen. Über Mailinglisten und Bekanntenkreis haben sich 32 Probanden bereiterklärt an der Studie teilzunehmen. Dabei waren 72 Prozent Männer und 28 Prozent Frauen. Das Alter der Probanden war zwischen 12 bis 54 Jahren vertreten und das Durchschnittsalter war 22 Jahre [Abbildung 6.2](#). Die Studie hat zwischen 30 Minuten und einer Stunde gedauert. Die Studien wurden an drei verschiedenen Orten durchgeführt, im TECO in Karlsruhe, in einem Seminarraum an der Hochschule Darmstadt und in einem Arbeitszimmer in Meschede.

Für jeden teilgenommenen Probanden wurde der gleiche Ablauf durchgeführt. Vor der Studie wurde ein Termin mit dem Probanden vereinbart. Nachdem der Proband zur abgemachten Zeit am vorgegebenen Ort angekommen ist, wurde ihm erklärt, wofür die Studie ist, was man mit der Studie herausfinden will und welche Erwartungen man an den Probanden hat. Als der Proband alles verstanden hat und die Einverständniserklärung verstanden und unterschrieben hatte, wurde ihm das Armband angezogen. Dabei haben die Rechtshänder das Armband an dem linken Arm und die Linkshänder an dem rechten Arm angelegt bekommen.

Da man für die Studie ein System mit einer Grafischen Oberfläche (GUI) entworfen hatte, wurde der Proband mithilfe des Systems durch die Studie geführt. Als Nächstes wurden dem Probanden ein paar Personalien abgefragt, wie das Alter, das Geschlecht, ob sich die Person als Musikalisch empfindet, ob man Computerspiele spielen würde, ob man schon einmal eine Smartwatch benutzt habe und ob die Person schon mal ein Taktiles Gerät benutzt habe. Falls vom Probanden Fragen während der Studie Fragen aufgekommen sind, wurden diese sofort beantwortet.

Nach der Aufnahme der Personalien, wurde dem Benutzer erklärt, was ihn als nächstes erwartet und von Ihm verlangt wird. Man hat dem Nutzer im ersten Schritt 10 Signale abgespielt, um Ihn ein Gefühl für Signale zu geben. Im Anschluss wurde dem Probanden jedes Signal einzeln erneut abgespielt. Dabei sollte er das Signal zu drei jeweiligen Kategorien zuordnen, die Kategorien waren die Signaltypen *Kurz*, *Mittel* und *Lang*. Dieser Schritt war dafür notwendig, um für den Benutzer die Grenzen für die jeweilige Kategorien zu bestimmen. Diese Grenzen sind für die Initialisierung des Algorithmus notwendig gewesen.

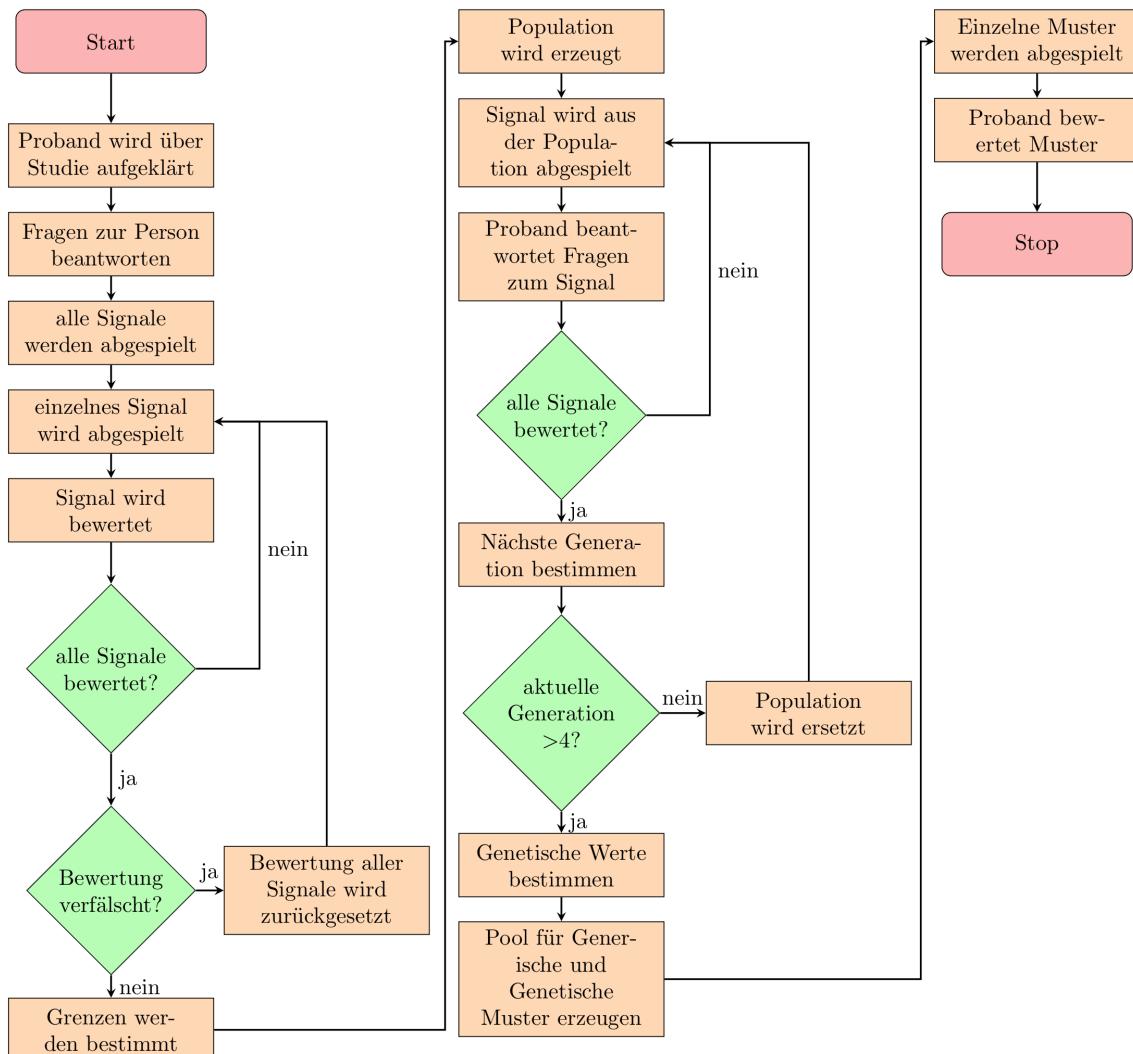


Abbildung 6.1: Studienablauf

Als die 10 Signale bewertet wurden, wurde der Benutzer darüber aufgeklärt, was Ihm als nächsten Schritt erwartet. Es wurde ihm ein anhand seiner vorherigen Eingaben eine Population von Signalen erzeugt, es wurde ein zufälliges Signal abgespielt, dass er bewerten sollte. Mittels drei Fragen wurde das Signal bewertet. Um eine Iteration komplett zu bewerten wurde dieser Vorgang 30 mal wiederholt. Im Anschluss wurde gefragt, wie der Benutzer sich derzeit fühlt. Nachdem dem der Benutzer eine Iteration komplett bewertet hat, wurden neue Werte berechnet. Es wurden insgesamt vier Iterationen durchgeführt, um einen möglichst genauen Wert für den Benutzer zu bestimmen.

Im letzten Schritt wurde der Benutzer aufgeklärt, dass ab dem Zeitpunkt nur noch Folgen von Signalen, die man ab jetzt Muster nennt, abgespielt werden. Die Probanden sollten angeben in welcher Reihenfolge, was für Signaltypen abgespielt wurden. Vor der Studie wurden alle Muster einmalig festgelegt, die jedem Probanden abgespielt wurden. Es gab zwei Arten von Muster, die generischen Muster und die genetischen Muster. Der einzige Unterschied zwischen den beiden Arten waren die Werte, die die Signale in einem Muster zugewiesen wurden. Das bedeutet es wurden zweimal dasselbe Muster abgespielt mit lediglich anderen Werten. Die genetischen Muster hatten die Werte, die nach dem Algorithmus erzeugt wurden übernommen, wobei die generischen Muster einen vordefinierten Wert zugewiesen bekommen hatten. Der generische Wert ist für jeden Probanden gleich gewesen und man hat sich wie im Entwurf schon beschrieben am Paper [PBB16] orientiert. Dabei gab es Muster mit drei, vier und fünf Signalen. Nacheinander wurde dem Nutzer zuerst alle Muster mit drei Signalen, gefolgt von allen Mustern mit vier und zuletzt alle Muster mit fünf Signalen abgespielt. Das genetische Muster wurde abwechselnd zum generischen Muster abgespielt.

Nachdem alle Muster von dem Probanden bewertet wurden, haben Sie Ihre e-Mail noch auf Papier angegeben, um an einer automatischen Verlosung von zwei Gutscheinen teilzunehmen. Bei Interesse wurde Ihnen Ihre Werte gezeigt und erklärt, was genau im Hintergrund passiert worden ist. Während der ganzen Studie standen dem Probanden ausreichend Süßigkeiten zur Verfügung, bei denen Sie sich frei bedienen konnten.

6.2 Analyse

6.2.1 Angaben der Probanden

Nach der Befragung der Probanden hat sich ergeben Abbildung 6.2, dass 31% musikalisch sind, 81% gelegentlich Computerspiele spielen. 21% haben schon mal eine Smartwatch benutzt und genau die Hälfte haben schon mal taktile Geräte benutzt.

Die Auswertung der Übertragung hat ergeben, dass es im Durchschnitt 300ms gedauert hat, bis das Signal übertragen worden ist.

6.2.2 Initialisierung der Grenzen

Für die Bestimmung der Grenzen von *Kurz*, *Mittel* und *Lang* hat der Benutzer 10 Signale zu der jeweiligen Kategorie einordnen sollen. Da man bei 16% der Probanden bei diesem Schritt keine eindeutigen Grenzen bestimmen konnte, musste der Schritt ein weiteres mal wiederholt werden.

Obwohl die 10 normalverteilten Signale in zufälliger Reihenfolge abgespielt wurden, wurde bei 84% aller Probanden die zehn Signale so bewertet, dass zuerst alle *Kurz*, gefolgt von nur *Mittel* und anschließend nur *Lang* bewertet wurden. Die restlichen Probanden haben an den Grenzen der Signaltypen nur *Kurz* und *Mittel* oder *Mittel* und *Lang* vertauscht.

In der Auswertung Abbildung 6.3 haben sich drei Normalverteilungen gebildet. Daraus kann man entnehmen, dass die Werte 100ms und 200ms eindeutig als *Kurz*, sowie auch die

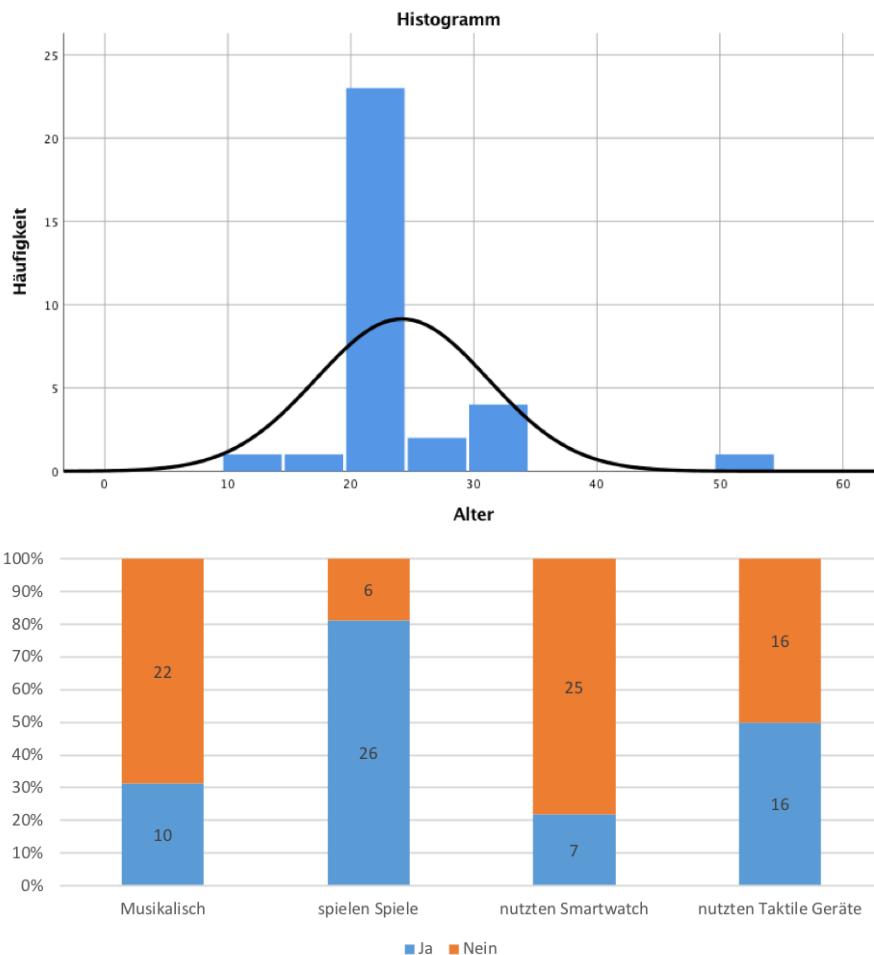


Abbildung 6.2: Alter aller Probanden (oben) und Auswertung der Fragen (unten)

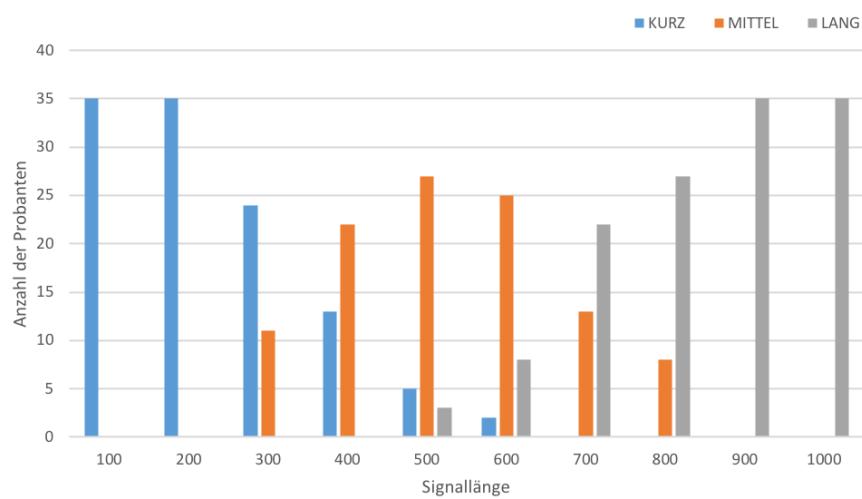


Abbildung 6.3: Auswertung der Initialisierung der Grenzen, Die Signalsträrke wird in ms angegeben

Werte 900ms und 1000ms als *Lang* erkannt worden sind. Einen eindeutigen Hochpunkt für Mittel hat man bei dem Wert von 500ms, dennoch gab es einige Probanden für die dieser Wert noch als *Kurz* oder *Lang* empfunden wurde.

Aus dem Diagramm kann man sehr schön erkennen, dass man eine Personalisierung von Vibrationen benötigen könnte, da man außer den Rändern keine eindeutige Zuweisung von *Kurz*, *Mittel* oder *Lang* unter allen Probanden bestimmen konnte. Man kann zwar im Vorfeld Werte für Signale definieren, wie es in dem Paper [PBB16] gemacht wurde. Bei mehreren Signallängen ist die Differenzierung von Personen unterschiedlich.

6.2.3 Auswertung der Iterationen des Algorithmus

Verlauf der Grenzen

Der Algorithmus wurde insgesamt viermal ausgeführt, anschließend wurde aus der letzten Population der personalisierte Wert gemittelt.

Kurz

Eine der größten Veränderungen im Diagramm Abbildung 6.4 gibt es bei *Kurz*; die oberste Grenze vom Maximum ist nach vier Iterationen von 600ms auf 415ms gesunken. Die obere Grenze des Minimums ist von 100ms auf 300ms gestiegen. Der Median von den Maximum und Minimum konvergieren in die gleiche Richtung. Nach vier Iterationen ist der Median vom Minimum um 80ms nach oben und der Median vom Maximum auch um 80ms nach unten gewandert. Das Minimum und Maximum überschneidet sich in der 2. Iteration nur minimal. In der 4. Iteration überschneiden sich Minimum und Maximum um 150ms. Das 50% Quartil vom Minimum wächst um bis zu 100ms, wobei das 50% Quartil beim Maximum sich das 200ms Intervall auf ein 120ms Intervall verkleinert. Das gesamte Intervall von *Kurz* verkleinert sich insgesamt um 44% von 100-600ms auf 140-420ms, somit ist das die größte Veränderung unter allen Intervallen.

Mittel

In der ersten Iteration besitzt man schon Überschneidungen beim Minimum und Maximum. Von der ersten zur zweiter Iteration verschieben sich die Grenzen des Minimums um ca. 10ms nach oben, diese werden im Verlauf der Iteration minimal kleiner. Die Grenzen des Maximums hingegen werden größer und verschieben sich anschließend minimal nach unten. Das 50% Quartil verkleinert sich um 40% beim Minimum von einem 200ms Intervall auf ein 120ms Intervall und verschiebt sich anschließend im Verlauf der folgenden Iterationen nur nach oben. Bei *Mittel* hat sich das gesamte Intervall um 16% von 300-800ms auf 345-766ms verkleinert.

Lang

Die meisten Ausreißer sind in bei *Lang* vertreten, hier hat man in der zweiten Iteration schon ein Maximum von 1000ms auf unter 800ms und 880ms erhalten, das ist eine Änderung eines Maximums um 200ms nach nur einer Iteration bei einem Probanden. Das 50% Quartile des Maximums werden minimal größer und wachsen nach unten. Die 50% Quartile des Minimums werden kurzzeitig größer, minimieren sich minimal im Verlauf. Die Grenzen des gesamten Intervalls von *Lang* haben sich um 11% von 500-1000ms auf 555-997ms minimiert. Zwischen der 3. und 4. Iteration bei *Lang* ist es das einzige mal vorgekommen, dass zwischen zwei Iterationen bis auf 3ms keine Unterschiede erkennbar sind. Das würde bedeuten, dass nach der 3. Generation für die Probanden schon ein sehr gut unterscheidbares Ergebnis vorhanden ist.

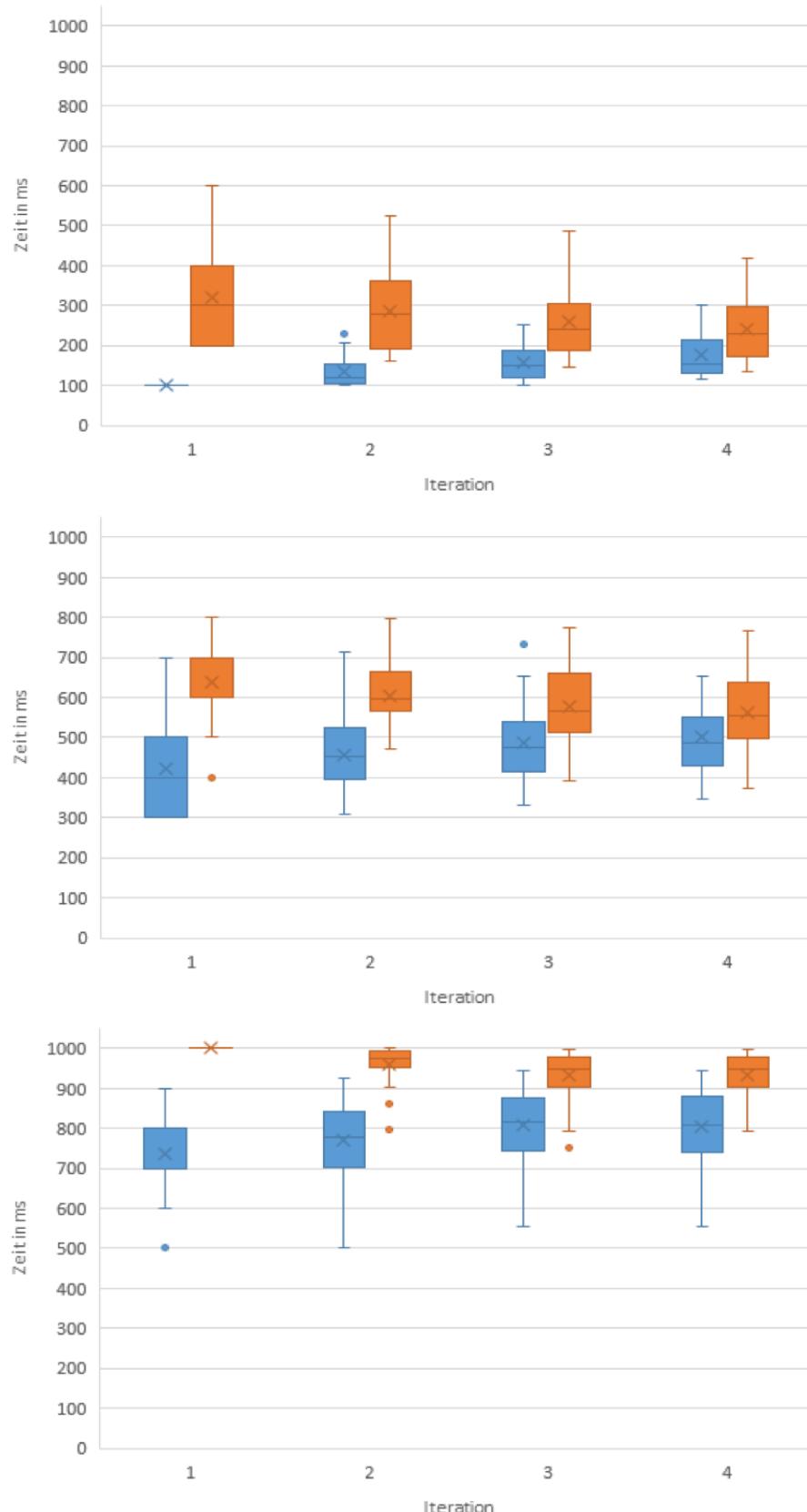


Abbildung 6.4: Darstellung des Minimum (in blau) und Maximum (in rot) im Verlauf der 4 Iterationen des Algorithmus. Das erste Bild zeigt *Kurz*, gefolgt von *Mittel* und *Lang*.

Unter den Signaltypen überschneiden sich die Grenzen selbst, diese verringern sich im Verlauf der Iterationen.

Das Minimum steigt stetig und das Maximum verringert sich. Denn genau diesen Effekt wollte man auch erzielen, dass der Benutzer zu seinen personalisierten Wert konvergiert. Bei *Kurz* und *Mittel* überschneiden sich die 50% Quartile. Bei *Lang* überschneiden sich diese in keiner Iteration.

Schlussfolgernd kann man sagen, dass die man anhand der 50% Quartile vom Maximum und Minimum aller Probanden einen Kurzen Signal in dem Bereich von 130ms bis 300ms, ein Mittleres Signal zwischen 430ms und 640ms und ein Langes Signal zwischen 740ms und 980ms gebildet haben.

Grenzen nach dem Algorithmus

In dem Diagramm Abbildung 6.5 sind alle gemittelten personalisierten Werte nach der vierten Iteration abgebildet. Daraus lässt sich ableiten, dass zwischen den Signaltypen von *Kurz*, *Mittel* und *Lang* sich die Grenzen überschneiden und man somit sagen kann, dass man diese Bereiche auf für vordefinierte Signale meiden sollte. Die Grenzen überschneiden sich zwischen *Kurz* und *Mittel* von 289-355ms und zwischen *Mittel* und *Lang* von 729-752ms. Für die einzelnen Signaltypen hat sich der Median für *Kurz* bei 202ms, für *Mittel* bei 548ms und *Lang* bei 893ms gebildet. Die oberen 25% von *Lang* und die unteren 25% von *Kurz* weisen nur eine minimale Abweichung auf. Im Gegensatz dazu gibt es eine Abweichung von genau 109ms bei den oberen 25% von *Kurz* und den unteren 25% von *Lang*. Es gibt einen Ausreißer bei *Lang* mit einem Wert von 635ms, dabei handelt es sich nicht um keinen Fehler, sondern einen Probanden der genau dies für sich als personalisierten Wert bestimmt hat. Im Vergleich zu dem inneren 50% Quartil von *Kurz*, das ein 90ms Intervall hat, und für *Lang* 99ms Intervall existiert, hat *Mittel* den Wert von 144ms. Damit ist *Mittel* um nahezu 50ms größer als die anderen Signaltypen.

Stimmung im Verlauf des Algorithmus

Nach jeder Iteration wurden die Benutzer gefragt, wie Ihre Stimmung Abbildung 6.6 aktuell sei. Dabei hat es bei 62% aller Probanden keine Änderung der Stimmung gegeben. Diese Probanden hatten Ihre Stimmung mit *Gut* oder *Sehr gut* bewertet. Im Gegensatz dazu hat sich die Stimmung bei 16% der Probanden entweder zwischen *Gut* und *Sehr gut* oder *Gut* und *OK* abgewechselt. Des Weiteren gab es bei 13% der Probanden eine aufsteigende Stimmung und bei den restlichen 9% ist die Stimmung pro Iteration gefallen.

Somit kann man feststellen, dass die Anzahl von vier Iterationen keinen wirklichen Einfluss auf die Stimmung des Benutzers ausgewirkt hat.

Replays eines Signals

Man hat den Probanden die Möglichkeit geboten, ein Signal wiederholen zu dürfen. Im Verlauf des Algorithmus wollte man herausfinden, wie oft unter allen Probanden auf Replay für ein bestimmtes Intervall gedrückt wurde Abbildung 6.7.

In der ersten Iteration haben 44% der Probanden in dem Intervall von 700-799ms die größten Schwierigkeiten gehabt ein Signal zuzuordnen und sich das Signal erneut abgespielt. Der Bereich, der am wenigsten wiederholt abgespielt werden musste war 200-299ms. Die restlichen Intervalle der ersten Iteration wurden mit einem Mittelwert von 32% erneut abgespielt.

In der zweiten Iteration hat sich die Anzahl der Replays um einiges verringert, hier war der Mittelwert insgesamt bei 22%, das Minimum war bei 200-299ms im Wert von 15% und

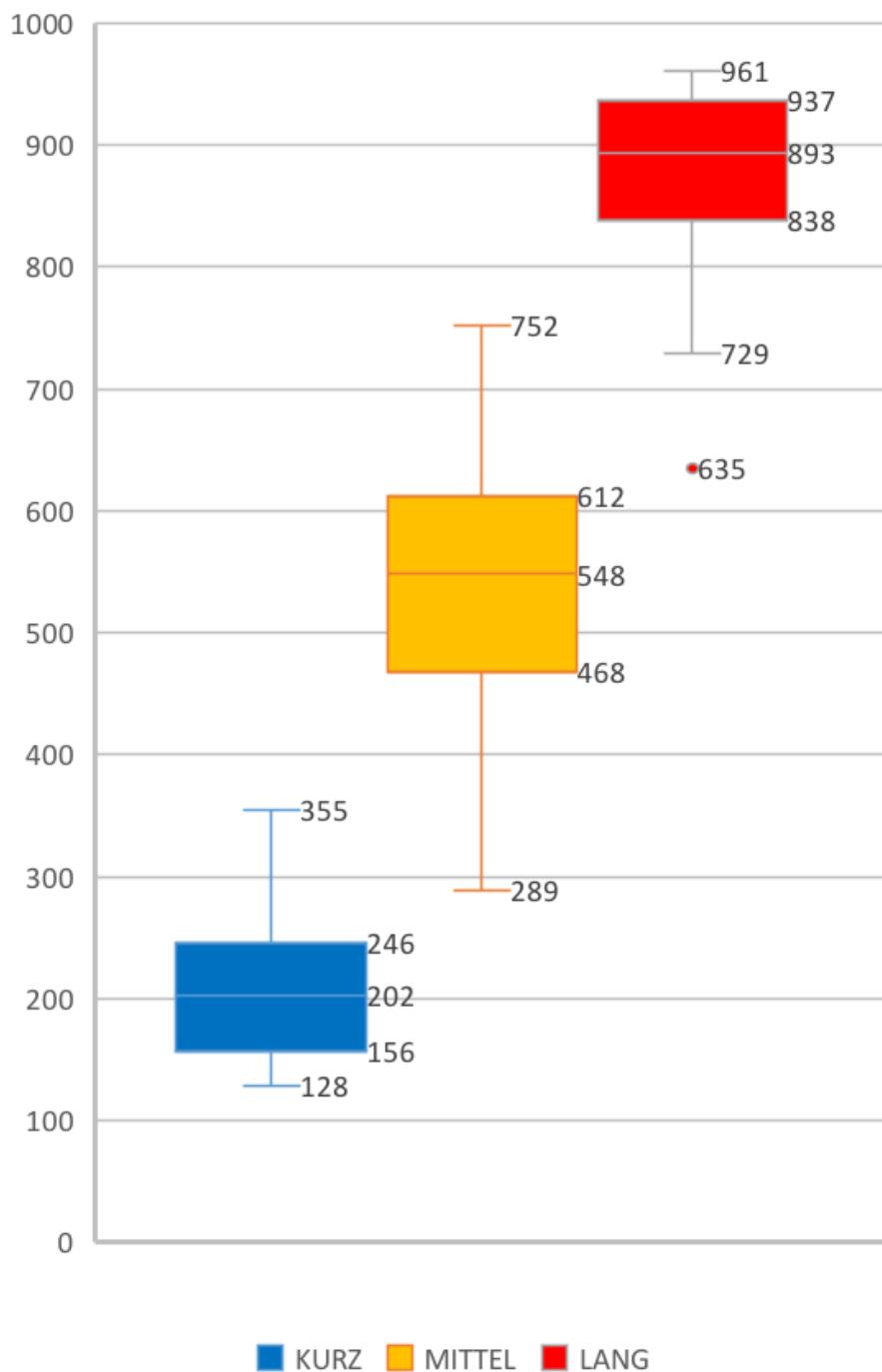


Abbildung 6.5: Nach der Vierten Iteration gemittelte personalisierte Werte in ms

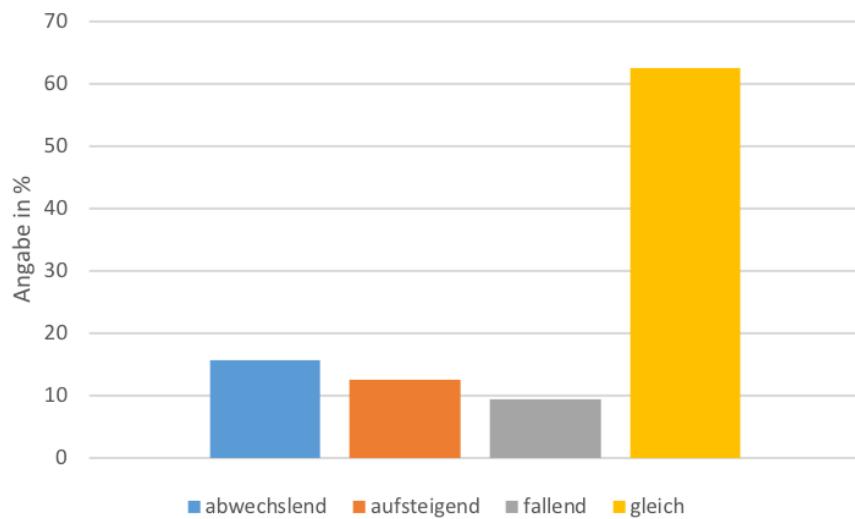


Abbildung 6.6: Bewertung der Stimmung

das Maximum bei 500-599ms mit 27%. Die dritte Iteration hat eine Verbesserung bei allen nahezu allen Werten gehabt, bis auf das Intervall von 700-799ms was auf 35% angestiegen ist; insgesamt ist der Mittelwert in dieser Iteration bei 19%.

Selbst in der letzten Iteration hat sich ein globales Minimum unter allen Iterationen bei 800-899ms von 9,5% gebildet. Das globale Maximum war bei 700-799ms in der ersten Iteration mit 43%, ist aber auch in der vierten Iteration, trotz Verbesserungen zwischen durch das Maximum mit 42,5% geblieben.

Man kann daraus erkennen, dass die Probanden genau an dem Bereich von Kurz zu Mittel, was sich in der 4. Iteration bei 200-399ms bewegt, und Mittel zu Lang, dass zwischen 600-799ms liegt, am meisten den Replay Knopf gedrückt haben. Ein viertel aller Probanden haben bei 900-1000ms in jeder Iteration das Signal erneut abgespielt. Jedoch kann man in der vierten Iteration sehr gut erkennen, dass die Signale mit den Werten 100-199ms, 400-599ms und 800-899ms am geringsten erneut abgespielt werden mussten. Im Durchschnitt wurde, der Replay Button 1,7mal gedrückt, wenn man ein Signal erneut abgespielt haben wollte. Dabei war das Minimum einmal und das Maximum neunmal, dass man das Signal erneut abspielen wollte.

Mittelwert

Im Verlauf des Algorithmus konnte man feststellen, dass zwischen der ersten und der letzten Iteration unter allen Probanden der Mittelwert sich wie im folgenden Verhalten hat. *Kurz* hatte eine durchschnittliche Abweichung von 10,11%, bei einem Minimum von 1% und einem Maximum von 23%. Weiterhin habe sich der Mittelwert bei den beiden Signaltypen *Mittel* und *Lang* von der ersten zur letzten Iteration im Durchschnitt nur um 3,34% verändert. Dabei war das Minimum bei 0,09% und ein Maximum bei 9,9% für *Mittel* sowie ein Minimum von 0,06% und ein Maximum von 15,33% für *Lang* vorhanden.

Beantwortung der Fragen

Im Verlauf des Algorithmus hat man auch die Zeit Abbildung 6.8, die benötigt wurde, um die Fragen für ein Signal zu beantworten, gespeichert. Daraus hat sich ergeben, dass *Kurz* in jeder Iteration um einer ganzen Sekunde schneller beantwortet wurde als die anderen beiden Signaltypen. Für *Mittel* und *Lang* hat man im Durchschnitt immer gleich viel Zeit benötigt. Nach der dritten Iteration ist der Wert für alle drei Signaltypen nahezu konstant

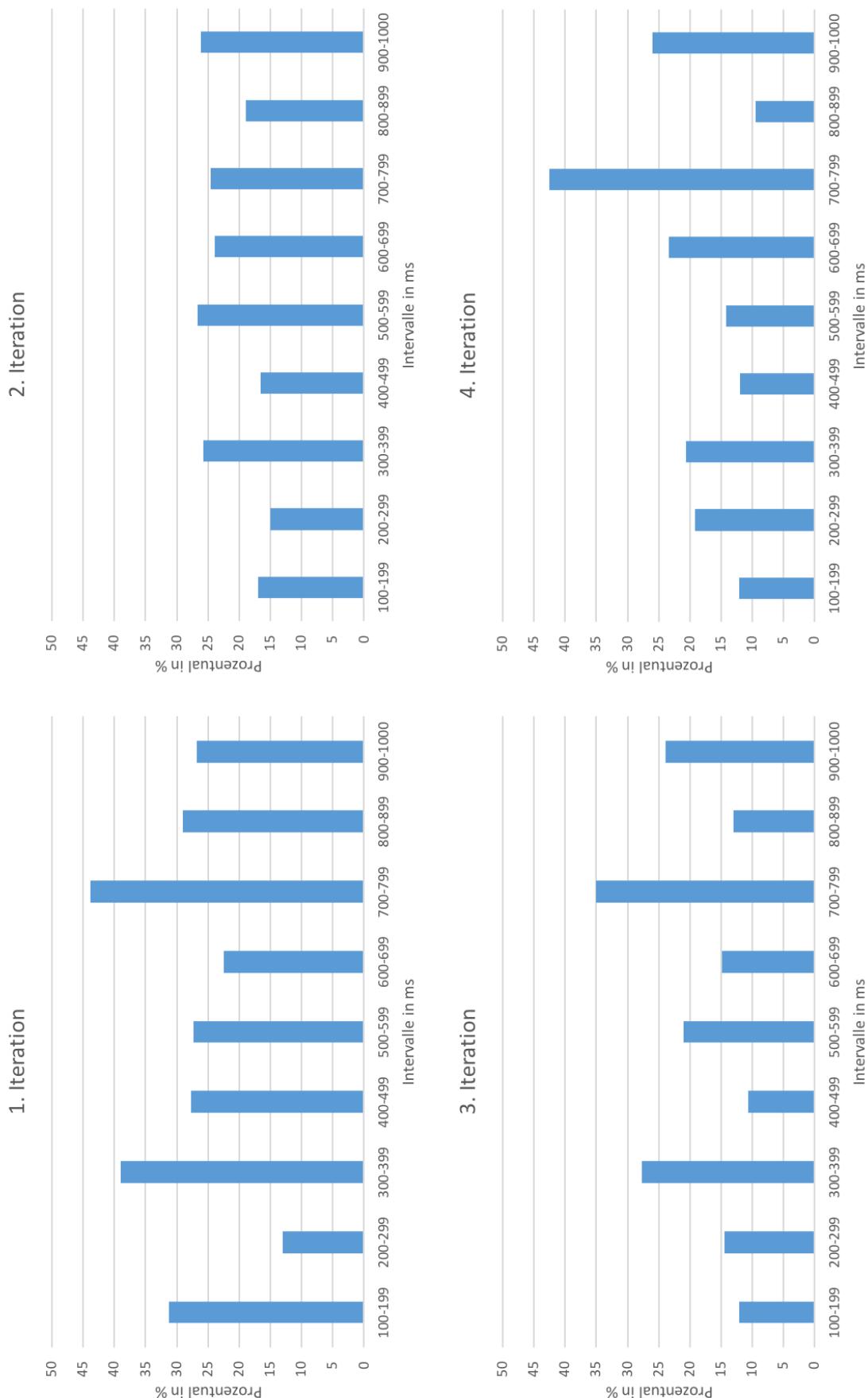


Abbildung 6.7: Anzahl der Replays innerhalb von Zeitintervallen über dem Verlauf des Algorithmus

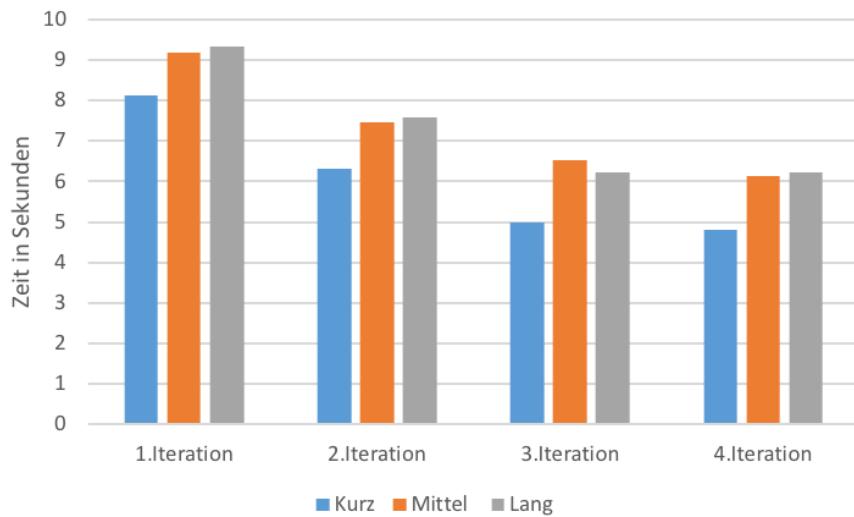


Abbildung 6.8: Durchschnittlich benötigte Zeit der Probanden, um die drei Fragen des Algorithmus zu bewerten.

geblieben, des weiteren hat sich bis zur dritten Iteration bei allen Signaltypen der Wert um drei Sekunden verringert. Daraus lässt sich ableiten, dass die Probanden sich an die Signale gewöhnt haben und die drei Fragen schneller beantworten konnten.

Verlauf der Werte

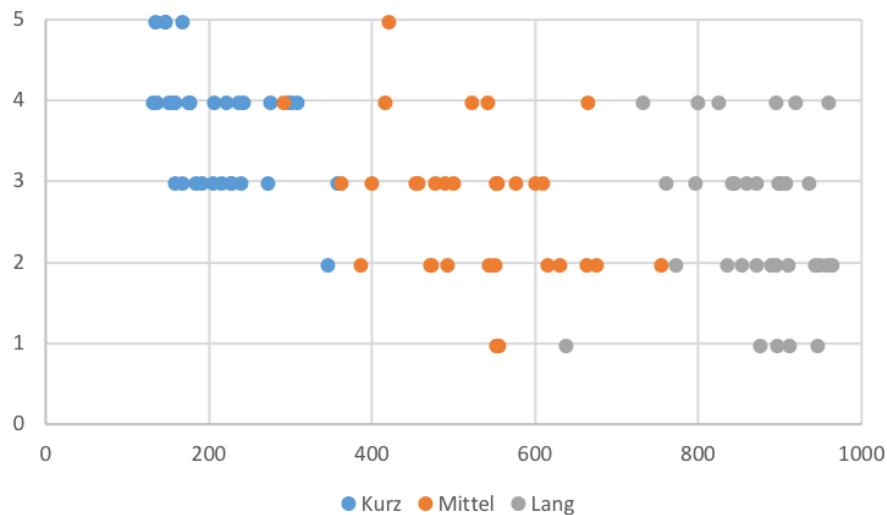


Abbildung 6.9: Personalisierte Werte der einzelnen Probanden, die in dem Diagramm der Stärke über die Zeit dargestellt werden.

Anhand des Diagramms Abbildung 6.9 kann man erkennen, dass die meisten Probanden ein stärkeres *Kurz* Signal sowie ein schwächeres *Lang* Signal bevorzugt haben. Für das Signal *Mittel* hat man sich jedoch in dem mittleren Stärkebereich befunden.

In der ersten Iteration Abbildung 6.10 erkennt man sehr gut, dass es noch keine wirkliche Unterteilung existiert. Bis zum Bereich von 250ms hat sich eindeutig *Kurz* und in dem Bereich von 800-1000ms eindeutig *Lang* gebildet. In dem restlichen Bereich existieren Überschneidungen der Signaltypen. Anhand der zweiten Iteration haben sich schon leichte Cluster gebildet, bei denen es dennoch Ausreißer von *Kurz* und *Mittel* an den Werten nahe

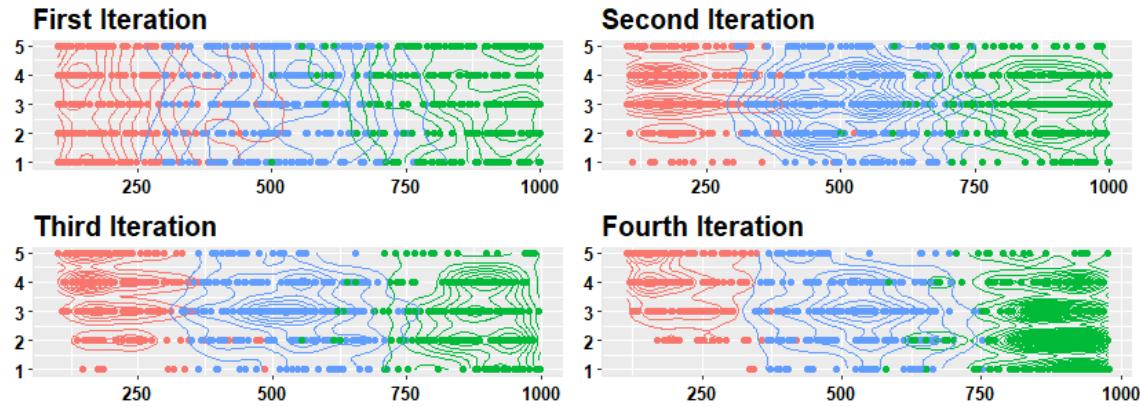


Abbildung 6.10: Darstellung der Stärke zu der jeweiligen Signallänge über dem Verlauf des Algorithmus. Dabei bildet rot *Kurz*, blau *Mittel* und grün *Lang* ab.

Tabelle 6.1: Durchschnittliche Genauigkeit zur Erkennung eines Signals der Probanden über die Kategorien (Angaben in %)

Genetisch			Generisch			Kategorie
3er	4er	5er	3er	4er	5er	
81	77	75	76	69	66	über alle Probanden
76	71	74	75	67	63	weiblich
83	79	77	80	74	71	männlich
80	74	77	79	74	71	musikalisch
80	75	72	77	69	66	nicht musikalisch
81	76	75	78	72	70	spielen Spiele
75	70	70	74	63	58	spielen keine Spiele
86	84	80	83	79	75	nutzten Smartwatch
78	72	72	76	68	66	nutzten keine Smartwatch
83	78	73	80	70	67	nutzten Taktiles Gerät
76	72	75	75	70	69	nutzten kein Taktiles Gerät

500ms existieren. Bis zur vierten Iteration verdichten sich die Signale von *Kurz* in dem oberen linken Bereich, das bedeutet, dass man für Signale vom Typ *Kurz* eher stärkere Signale bevorzugt hat. Die Signale des Typs *Mittel* haben im Verlauf des Algorithmus genau in der Mitte bei dem Wert 500ms und der mittleren Stärkestufe einen Hochpunkt gebildet. Des weiteren hat sich das Signaltyp *Lang* in den Stärkestufen eins (für sehr schwach) bis drei (für o.k.) einzelne Cluster gebildet. Wohingegen bei *Kurz* und *Mittel* sich ein Hochpunkt gebildet haben.

Man kann zweifellos erkennen, dass wenn man noch weitere Iterationen ausgeführt hätte, würde sich das Ergebnis weiter verdichten.

6.2.4 Muster

Die Tabelle 6.1 beschreibt, wie genau die Probanden ein Signal aus den Mustern erkannt haben, man hat die generischen Muster mit den genetischen Mustern verglichen, sowie für jede Gruppierung von Probanden die Auswertung neu bestimmt.

Laut der Tabelle kann man feststellen, dass die Probanden die keine Computerspiele spielen, die einzelnen Signale der generischen fünf Muster nur zu 58% richtig erkannt haben. Dieser Wert ist das globale Minimum unter allen Gruppierungen der Probanden. Bei den

Probanden, die schon einmal eine Smartwatch nutzten, haben bei den genetischen dreier Mustern ein einzelnes Signal mit 86% Wahrscheinlichkeit richtig erkannt, der auch als globales Maximum unter allen Probanden-Gruppen existiert.

Unter den vorliegenden Ergebnissen haben die männlichen Probanden im Vergleich zu den weiblichen Probanden die einzelnen Signale aller Muster-Typen besser erkannt. Bei den dreier Mustern können die musikalischen im Vergleich zu nicht musikalischen Probanden die Signale gleich gut erkennen, jedoch zeigen die musikalischen Probanden bei vierer und fünfer Muster eine Verbesserung bis zu 5%. Allgemein können die gelegentlich Computerspiele spielenden Probanden im Gegensatz zu nicht-Spielern alle abgespielten Signale der Muster-Typen besser zuweisen. Wenn wir ins Detail gehen, liegt bei den generischen fünf Muster die Abweichung von 12%. Dasselbe Ergebnis, nämlich 12% Unterschied kommt bei Probanden vor, die schon mal eine Smartwatch getragen haben.

Bei der letzten Gruppe wird es interessant, denn hier haben genau 50% der Probanden schon mal ein Taktiles Gerät verwendet. Dabei wurde um 8% der Signale der genetischen dreier Muster sowie um 4% der generischen Muster von den Teilnehmern, die Erfahrung mit den taktilen Geräten hatten, besser erkannt. Bei den Signalen der fünf Muster haben die Probanden, die schon einmal ein Taktiles Gerät verwendet haben, die Signale um 2% besser erkannt.

Im Vergleich zu den generischen Muster wurden die Signale der genetischen Muster öfter richtig zugeordnet. Die Genauigkeit erreichte sogar bei dem genetischen dreier Muster bis zu 82%. Somit ist es um 5% besser als der Wert von den generischen dreier Mustern. Obwohl der Prozentsatz bei den vierer Mustern insgesamt gesunken ist, gewinnt jedoch der Unterschied an Ausdehnung. Hier erhöht sich das Optimierungspotenzial bei genetischen vierer Mustern um 8%. Generell wurden 75% der Signale bei den genetischen fünf Muster korrekt erkannt und somit wird nur zwei Drittel der Signale von den generischen fünf Muster als richtig empfunden.

In der Darstellung [Abbildung 6.11](#) wurden die Muster und die Häufigkeit, wie oft die Probanden das Muster nicht als solches erkannt haben, dargestellt. Die Muster *KKK* und *LLL* haben die meisten Probanden korrekt erkannt, nur 12 % haben bei den genetischen Werten das Signal nicht korrekt erkennen können. Außerdem hat man bei denselben Mustern mit den generischen Werten eine Verschlechterung von 7% bei *KKK* und eine Verdopplung bei *LLL* festgestellt, wobei man das gleiche Verhalten auch bei den vierer Muster erkennen kann.

Wenn man die drei Diagramme vergleicht, erkannt man, dass für die dreier und vierer Muster mit nur einem Signaltypen (*Kurz* und *Lang*) nahezu identisch sind. Die genetischen Muster wurden dabei öfter erkannt als die generischen Muster. Das Muster mit durchgehenden *Mittel* Signalen haben die Probanden schlechter erkannt als die anderen beiden Signaltypen und es waren bei den dreier sowie bei den vierer Mustern jeweils das genetische um das doppelte besser als die generischen Muster.

Unter allen dreier Mustern wurden zwei Drittel der genetischen Muster besser als die generischen Muster erkannt. Zusätzlich befindet sich der Durchschnitt bei 36% für die genetischen und bei 43% für die generischen dreier Muster. Dieser Prozentsatz steigt bei den genetischen vierer Muster auf 46% und bei den generischen auf 50%, bei den fünf ist der Wert bei 57% für die genetischen und bei 64% für die generischen Muster. Anhand dieser Daten kann man sagen, je höher die Anzahl der Signale in einem Muster werden, desto weniger Probanden können diesen noch richtig erkennen.

Des Weiteren kann man feststellen, dass die genetischen vierer Muster nur noch eine minimale Abweichung zu den generischen Muster aufweisen, dasselbe ist auch bei den fünf Mustern zu erkennen. Durch dem Erhöhen der Komplexität hat sich nicht alles

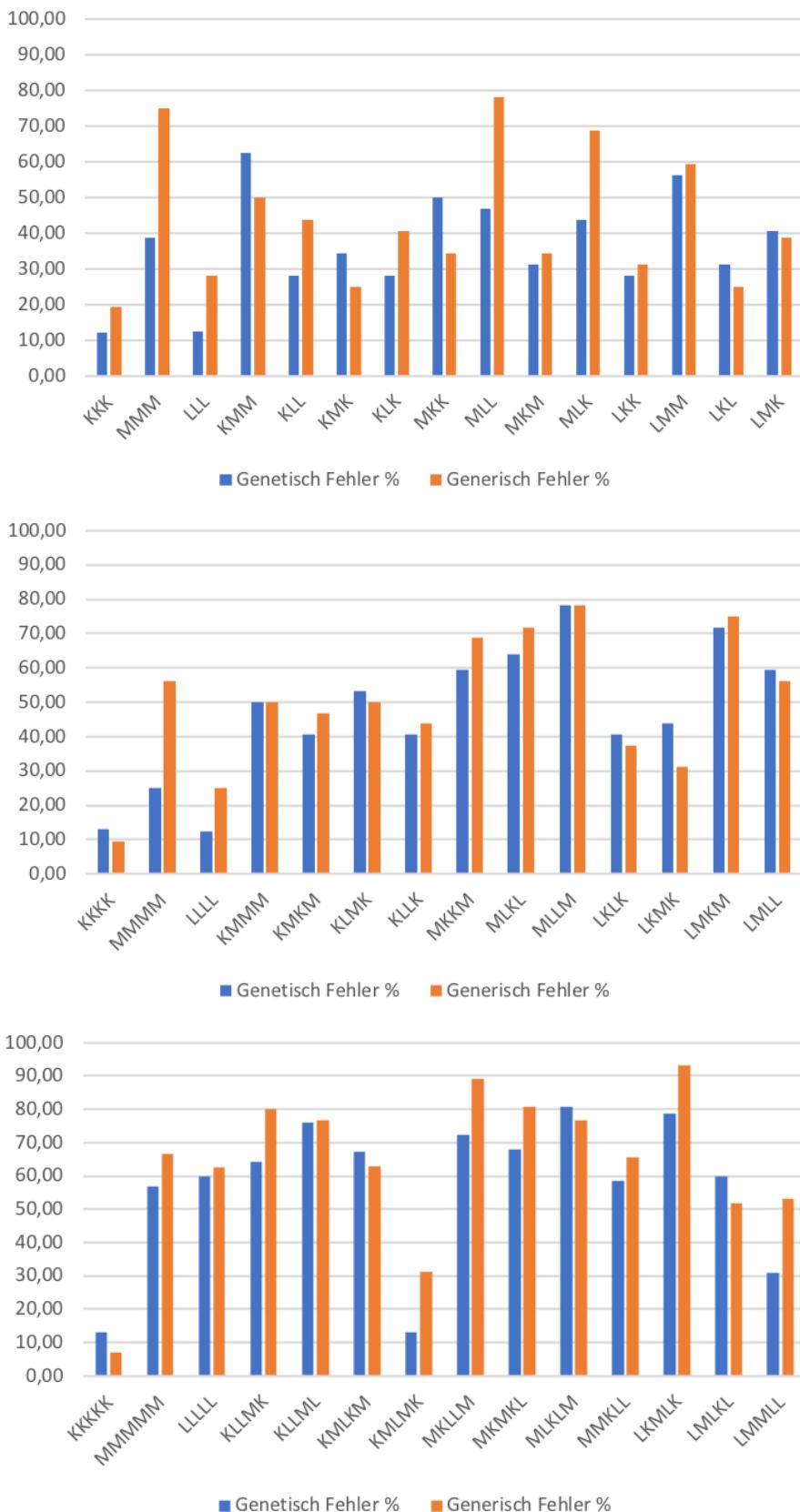


Abbildung 6.11: Anhand der Diagramme wird die Häufigkeit, wie oft einem Muster nicht richtig erkannt worden ist dargestellt. Es zählt ein Muster als nicht erkannt, wenn mindestens ein Signal nicht korrekt zugrordnet werden konnte. Dabei wurden die genetischen und generischen Fehlerangaben (in %) direkt miteinander verglichen. Die Diagramme wurden für die jeweiligen Muster (dreier, vierer und fünfer) separat dargestellt. Hier steht K für *Kurz*, M für *Mittel* und L für *Lang*.

Tabelle 6.2: Erkennungsrate (und Standardabweichung) der Mustertypen. Dabei wurden die Genetischen und Generischen Signale miteinander verglichen

	Genetisch	Generisch
3 Vibratoren	0,817 (0,135)	0,786 (0,106)
4 Vibratoren	0,771 (0,148)	0,729 (0,147)
5 Vibratoren	0,756 (0,123)	0,686 (0,138)

verschlechtert, denn bei dem auf- und wieder absteigenden Muster *KMLMK* haben die Probanden es genauso gut erkannt, wie das durchgängige Kurze fünfer Muster.

Die meisten Probanden haben bei den Mustern nur ein Signal falsch erkannt oder haben es um einem Signaltypen höher oder niedriger bewertet, dennoch haben die Probanden erkannt, dass die Signale sich verändert haben. Als Beispiel haben Sie das Muster *LMMLL* als *MKKMM* bewertet.

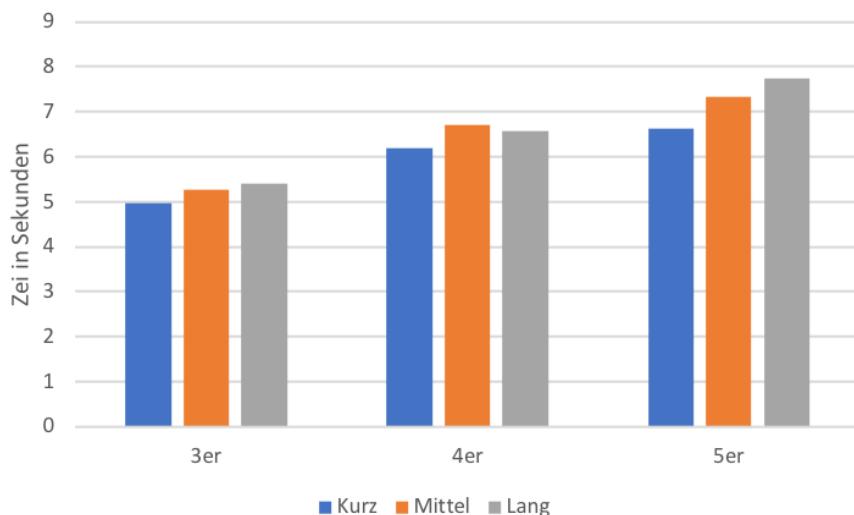


Abbildung 6.12: Durchschnittlich benötigte Zeit der Probanden, um ein Signal eines Musters zu erkennen

Im Gegensatz zu der benötigten Zeit beim Algorithmus [Abbildung 6.8](#), hat man für jeden Muster-Typen, der ein weiteres Signal hinzugefügt hat, mehr Zeit benötigt, um ein Signal zu erkennen [Abbildung 6.12](#). Die Zeiten für die Muster mit drei Signalen haben sogar eine bessere Zeit als die letzte Iteration vom Algorithmus. Alle Signaltypen bei dem dreier Muster haben eine ähnliche durchschnittliche Erkennungszeit von fünf Sekunden. Man kann gut erkennen, dass jeder Signaltyp um eine Sekunde stets wächst, bei wachsender Signallänge.

Mittels einer Varianzanalyse habe man die Werte aus der [Tabelle 6.2](#) bestimmt. Es hat sich ergeben, dass die Probanden signifikant schlechter abgeschnitten sind, als man sie mit einem komplexeren Muster konfrontiert hat, im Vergleich zu weniger komplexeren Mustern. Eine weitere Erkenntnis ist gewesen, dass die genetischen Muster besser erkannt wurden als die generischen Muster.

7. Zusammenfassung und Ausblick

Die Hypothese „personalisierte Vibrationen werden besser als vorgegebene Vibrationen erkannt“ wird im Folgenden beantwortet. Im Rahmen dieser Bachelorarbeit wurde ein System zur Bestimmung von personalisierten Vibrationsmustern entwickelt. Für das System habe man die Komponenten für einen Evolutionären Algorithmus entworfen und implementiert. Die vier Phasen des Systems wurden mit 32 Probanden durchgeführt.

Man darf auch nicht unerwähnt lassen, dass die Probanden niemals eine Rückmeldung erhalten haben, dass Sie das Signal richtig oder falsch bewertet hatten. Die Zeit der Bewertung sowie die Genauigkeit stieg pro Iteration des Algorithmus an. Im Anschluss des Algorithmus hat man die genetischen Vibrationssignale gegenüber dem generischen Vibrationssignalen bewerten lassen. Die Ergebnisse zeigen, dass die personalisierten Vibrationssignale signifikant besser erkannt worden sind, als die generischen Vibrationssignale. Während des Algorithmus haben die vier Iterationen kaum Auswirkungen auf die Stimmung gehabt. Die größte Schwierigkeit, die die Probanden hatten, waren die Grenzen zwischen den Signaltypen zu definieren. Somit hat man an den Grenzen ein vermehrtes Aufkommen der Wiederholungen eines Signals.

Man kann feststellen, dass je mehr Signale man in ein Muster besitzt, desto höher wird die Wahrscheinlichkeit, dass die Muster nicht korrekt erkannt werden. Bei einigen Probanden war unter der Betrachtung einer kleinen Abweichung der Mittelwert der bestimmten Anfangsgrenzen der personalisierte Wert. Die Genauigkeit hat sich mit der Erhöhung der Komplexität für leicht zu merkende Muster nicht verschlechtert.

Da die Studie nur mit einer Anzahl von 32 Probanden durchgeführt wurde, kann man diese Ergebnisse der Gruppierungen nur mit Vorbehalt betrachten, da es diese Ergebnisse nicht einfach auf eine größere Anzahl an Personen schließen kann. Somit heißt es nicht, dass Personen die weiblich sind immer schlechter Signale erkennen als männliche Personen. Wenn man dies überprüfen wollen würde, müsste man die Studie auf eine größere Anzahl an Probanden erweitern. Diese Werte sind nur die Ergebnisse, die aus meiner Studie dabei herausgekommen sind.

Anhand der in dieser Arbeit erkannten Erkenntnisse kann man für weitere Forschungen verwenden, indem man die Zeit der einzelnen Phasen weitestgehend verringern würde. Mittels dem Evolutionären Algorithmus muss man eine die komplette Population für eine Iteration bewerten lassen, dies hat zur Folge, dass man auf eine hohe Anzahl an Bewertungen gelangen kann und dementsprechend viel Zeit in Anspruch nehmen kann. Als alternative kann man es mit einem schneller konvergierendes Lösungsverfahren probieren.

Des Weiteren kann man anhand von optimierten Evolutionären Algorithmen versucht werden eine schnellere Konvergenz der personalisierten Werte zu erhalten.

Man sieht jedoch anhand dieser Arbeit, dass die Hypothese mit signifikanten Werten bestätigt wurde.

Ich bin der Meinung, dass man die personalisierten Vibrationen nicht außer Acht lassen sollte, da man sieht, das selbst große Hersteller noch keine optimalen Lösungen haben bieten.

8. Anhang

Im folgenden sind die Papierskizzen, die beim Entwurf der GUI benutzt wurden, dargestellt.

Angaben zur Person

Geschlecht	Männlich <input checked="" type="radio"/>	Weiblich <input type="radio"/>
Alter	<input type="text"/>	
	✓	
<input type="button" value="Bestätigen"/>		

Ream Spielt man viel PC? ✓
 ↗ Fühlen Sie sich musikalisch?
 ↗ konformieren

- Speichern in Magie ✓
- ↖ Darauf folge ✓
- ↖ in Reihenfolge ✓
- Eine Packwörter
- Frage Haben Sie schon Tastatur Geräte verwendet? ✓
- Haben Sie Schon mal Pulsschlag Smartwatches gehabt? ✓
- Falls ja, konnten Sie die Vibrationen selbst einstellen? ✓

Status oder initSignalPage

Bitte bewerten Sie das (gerade abgespielten) Signal.

X Startposition 1.

Kurz	Mittel	Lang
<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Layout	✓	1. Popup & Erklärung ✓
Main Sprechen	✓	2. Positionen ✓
In Reihenfolge	✓	3. Prüfen ob Baud verbinden und ansetzen
Replay Button	✗	
Testen		
verbinden		

- Text (Erklärung) anzeigen
 ↗ Cursor Bewegen auf Startposition
 - Signal abspielen
 ↗ Status bewerten
 - Zeit starten ✓
 - Bewegen ✗ ✓
 - Zeit stoppen ✓
 - Kurzer Delay ✗
 ↗ Bildschirm leer Speicher ✓

Bitte bewerten Sie das Signal. AlgSignalPage

X

✓ ① Was für einen Signaltypen haben Sie erkannt?

Kurz Mittel Lang

Kurz Mittel Lang

✓ ③ Wie empfanden Sie die Stärke des Signals?

zu schwach passend zu stark

zu schwach passend zu stark

✓ ② Wie gut haben Sie das Signal als kurzes Signal erkannt?

sehr schlecht sehr gut

sehr schlecht sehr gut

✓ replay

✓ Todo Signal.cs
create StringAlgo Signal()

Stärke	✓
Layout	✓
Daten Speichern	✓
Replay Button	✓
Reihenfolge	✓
Testen	✓
Vorbinden	✓
play	✓ 1

Wie gelangweilt sind Sie? ErmüdungPage

✓

✓ or

Play Reihenfolge

- Erklärungstext anzeigen ✓
- Cursor an position Bring ✓
- Cursor an position Bringen
- Signal abspielen ✓
- Zeit starten ✓
- Bewerten ✓
- speichern → Zeit Stoppen ✓
- Algo aufstellen - Kurzes Delay ✓
- Bildschirm Leeren ✓

Layout ✓

Daten einlesen ✓

In Main Speichern ✓

In Reihenfolge ✓

Speichern ✓

3/1
2)
127

Erkennung Page

Sie haben eine Folge von Signalen abgespielt bekommen.

Was für Signale haben Sie erkannt?

K - M - L - K - L ✓ Neues Intervall bestimmen.
 generierte / standard Population ✓ Kurz : Beginn - Ende & Stärke MinMax
 Mittel ✓ Mittelwert ✓ Mittel : Beginn - Ende & Stärke MinMax
 Lang ✓ Lang : Beginn - Ende & Stärke MinMax
 (Komplex) ✓ Signal erzeugung S Signale
 Layout ✓ hauptschwanger

Arduino - jeder hat eigene Stärke ~

Nach jeder Iteration speichern → Speicher Design anpassen;
 übersichtlicher &

Wenn Blie abrist, wie soll man einsetzen verbinden Auswahl zu beginn des Programms machen
 in welche Datei gespeichert werden soll kleine Daten

Nicht unbedingt 2. Datei auswählen für die großen Daten

Literaturverzeichnis

- [app18] *Apple Watch Settings*, 2018 (accessed May 1, 2018). <https://support.apple.com/de-de/HT204793>.
- [bro5] *Brockhaus-Enzyklopädie online*, 2005-. <http://gso.gbv.de/DB=2.2/PPNSET?PPN=517415410>, Druckausgabe u.d.T.: Brockhaus. Enzyklopädie in 30 Bänden (2006).
- [BTT05] David Benyon, Phil Turner und Susan Turner: *Designing interactive systems: People, activities, contexts, technologies*. Pearson Education, 2005.
- [Dil17] Zoeller Dillmann: *Maschinelles Lernen*, 2017.
- [Fli] Patrick Flick: *Evolutionäre Algorithmen*.
- [FSDS16] Christopher B Fleizach, Eric Taylor Seymour und Joel M Lopes Da Silva: *Custom vibration patterns*, jul # -5" 2016. US Patent 9,383,820.
- [GOS01] Francine Gemperle, Nathan Ota und Dan Siewiorek: *Design of a wearable tactile display*. In: *Wearable Computers, 2001. Proceedings. Fifth International Symposium on*, Seiten 5–12. IEEE, 2001.
- [Hol92] John Henry Holland: *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.
- [Hün07] Christoph Hünermann: *Brockhaus Enzyklopädie online*, 2007.
- [Las] Christian WG Lasarczyk: *Genetische Programmierung einer algorithmischen Chemie*.
- [PB03] Peter Parente und Gary Bishop: *BATS: the blind audio tactile mapping system*. In: *Proceedings of the ACM Southeast regional conference*, Seiten 132–137. Citeseer, 2003.
- [PBB16] Erik Pescara, Michael Beigl und Matthias Budde: *RüttelFlug: a wrist-worn sensing device for tactile vertical velocity perception in 3d-space*. In: *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, Seiten 172–175. ACM, 2016.
- [Sel03] Bianca Selzam: *Genetische Algorithmen*. Dortmund: TU Dortmund, 2003.
- [Shi12] Daniel Shiffman: *The Nature of Code: Simulating Natural Systems with Processing*. Daniel Shiffman, 2012.
- [Wei15] Karsten Weicker: *Evolutionäre algorithmen*. Springer-Verlag, 2015.

