



TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Interaktionskonzept zur Durchführung und Autorisierung von Zahlungen im Bereich Automotive auf Basis eines Picture Passcode über ein multimodales Conversational Interface

Bachelorarbeit

vorgelegt von:	Inga Glotzbach
Studiengang:	Media Engineering
Fakultät:	Elektrotechnik Feinwerktechnik Informationstechnik
Matrikelnummer:	2985250
Erstgutachter:	Prof. Dr. Oliver Hofmann
Zweitgutachter:	Prof. Dr. Heinz Brünig
Betreuer:	Steffen Blümm
Abgabedatum:	31.05.2020



Abstract

In einer schnelllebigen und vernetzten Welt wird erwartet, dass jederzeit und überall Zahlungen veranlasst werden können. Dies gilt auch für Zahlungen während der Fahrt. Dazu ist eine Schnittstelle erforderlich, welche die Initiierung einer Zahlung unterstützt, ohne die primäre Hauptaufgabe des Fahrens zu beeinträchtigen. Die vorliegende Bachelorarbeit untersucht ein Konzept zur Autorisierung einer Zahlung, auf Basis eines Picture Passcode und dessen Interaktion über ein multimodales Conversational Interface. Dabei wird eine prototypische Umsetzung des Conversational Design und einer Fahrsimulation mit Hilfe eines Prototyping Tool umgesetzt um die Funktionalitäten des Picture Passcodes zu testen. Ziel dieser Arbeit ist eine Autorisierungsmethode zu entwickeln die technische Aspekte in Bezug der Bedienbarkeit und Datensicherheit im Kontext Fahren erfüllt.



Erklärung



TECHNISCHE HOCHSCHULE NÜRNBERG
GEORG SIMON OHM

Prüfungsrechtliche Erklärung der/des Studierenden

Angaben des bzw. der Studierenden:

Name: Glotzbach Vorname: Inga Matrikel-Nr.: 2985250

Fakultät: Elektro-, Feinwerk-, Informationstechnik Studiengang: Media Engineering

Semester: Sommersemester 2020

Titel der Abschlussarbeit:

Interaktionskonzept zur Durchführung und Autorisierung von Zahlungen im Bereich Automotive auf Basis eines Picture Passcode über ein multimodales Conversational Interface

Ich versichere, dass ich die Arbeit selbständig verfasst, nicht anderweitig für Prüfungszwecke vorgelegt, alle benutzten Quellen und Hilfsmittel angegeben sowie wörtliche und sinngemäße Zitate als solche gekennzeichnet habe.

Ort, Datum, Unterschrift Studierende/Studierender

Erklärung zur Veröffentlichung der vorstehend bezeichneten Abschlussarbeit

Die Entscheidung über die vollständige oder auszugsweise Veröffentlichung der Abschlussarbeit liegt grundsätzlich erst einmal allein in der Zuständigkeit der/des studentischen Verfasserin/Verfassers. Nach dem Urheberrechtsgesetz (UrhG) erwirbt die Verfasserin/der Verfasser einer Abschlussarbeit mit Anfertigung ihrer/seiner Arbeit das alleinige Urheberrecht und grundsätzlich auch die hieraus resultierenden Nutzungsrechte wie z.B. Erstveröffentlichung (§ 12 UrhG), Verbreitung (§ 17 UrhG), Vervielfältigung (§ 16 UrhG), Online-Nutzung usw., also alle Rechte, die die nicht-kommerzielle oder kommerzielle Verwertung betreffen.

Die Hochschule und deren Beschäftigte werden Abschlussarbeiten oder Teile davon nicht ohne Zustimmung der/des studentischen Verfasserin/Verfassers veröffentlichen, insbesondere nicht öffentlich zugänglich in die Bibliothek der Hochschule einstellen.

Hiermit genehmige ich, wenn und soweit keine entgegenstehenden Vereinbarungen mit Dritten getroffen worden sind,
 genehmige ich nicht,

dass die oben genannte Abschlussarbeit durch die Technische Hochschule Nürnberg Georg Simon Ohm, ggf. nach Ablauf einer mittels eines auf der Abschlussarbeit aufgebrachten Sperrvermerks kenntlich gemachten Sperrfrist

von 0 Jahren (0 - 5 Jahren ab Datum der Abgabe der Arbeit),

der Öffentlichkeit zugänglich gemacht wird. Im Falle der Genehmigung erfolgt diese unwiderruflich; hierzu wird der Abschlussarbeit ein Exemplar im digitalisierten PDF-Format auf einem Datenträger beigefügt. Bestimmungen der jeweils geltenden Studien- und Prüfungsordnung über Art und Umfang der im Rahmen der Arbeit abzugebenden Exemplare und Materialien werden hierdurch nicht berührt.

Ort, Datum, Unterschrift Studierende/Studierender



Inhaltsverzeichnis

Abkürzungsverzeichnis	III
Abbildungsverzeichnis	V
1 Einleitung	1
1.1 Ausgangssituation	1
1.2 Zielsetzung	1
1.3 Vorgehensweise	2
1.4 Projektträger	2
2 Voraussetzungen zum Bezahlen während dem Fahren	4
2.1 Bisherige Systeme	4
2.2 Interfaces	6
2.3 Authentifizierung und Autorisierung	8
2.3.1 Wissensbasierte Verfahren	8
2.3.1.1 Persönliche Identifikationsnummer	9
2.3.1.2 Alphanumerische Passwörter	9
2.3.1.3 Grafische Passwörter	10
2.3.2 Besitz	15
2.3.3 Biometrische Verfahren	16
2.4 Ergonomie im Fahrzeug	18
2.4.1 Verkehrspychologie	18
2.4.2 Wahrnehmungspychologie	20
2.4.3 Aufmerksamkeit	21
3 Konzeption	23
3.1 Value Proposition Canvas	23
3.2 Konzeptentwicklung des Picture Passcode	27
3.2.1 Ausgangspunkt Blackberry Picture Password	27
3.2.2 Konzeptionsphase	29



Inhaltsverzeichnis

3.2.3	Funktionsweise des Picture Passcode	32
3.3	Ergonomie	36
3.3.1	Interfaces	36
3.3.2	Gestalterische Aspekte	37
3.4	Datensicherheit	38
4	Prototyping	40
4.1	Erstellung eines Prototypen	40
4.1.1	Fahrsimulation	41
4.1.2	Conversational Design	47
4.2	Usability-Test	51
4.2.1	Testaufbau	52
4.2.2	Nasa TLX	55
4.2.3	Evaluation	55
4.3	Funktionale Komponenten	59
5	Schlussbetrachtung	62
5.1	Zusammenfassung	62
5.2	Fazit	63
5.3	Ausblick	63
Glossar		65
Literaturverzeichnis		68
A Anhang		i
A.1	Vollständiges Storyboard der Konversation	i
A.2	Mock-Up-Screens des Prototypen	iii
A.3	Aufgabenstellung und Szenario für den Usability-Test	v
A.4	Ergebnisse und Gruppenteilung der Umfrage für den Usability-Test .	vi
A.5	Einverständniserklärung zur Audio-Aufnahme der Probanden	vii
A.6	Ergebnisse des NasaTLX	viii
A.7	Plakat zur Raumgestaltung der Usability-Testessen	x
A.8	Fragebogen zum Usability-Test	xi
A.9	Quellcode der Passcode-Generierung	xii



Abkürzungsverzeichnis

AIFF Audio Interchange File Format

CFT Conversational Flow Tool

CPT Conversation Prototyping Tool

CUI Conversational User Interface

DAS Draw-a-Secret

FAR False Acceptance Rate

FAS Fahrerassistenzsystem

FFR False Rejection Rate

FIS Fahrerinformationssystem

GPS Global Positioning System

GUI Graphical User Interface

HDMI High Definition Multimedia Interface

HiFi High-Fidelity

HTTPS Hypertext Transfer Protocol Secure

JS JavaScript

JSON JavaScript Object Notation

JSUI Javascript User Interfaces and Graphics

LoFi Low-Fidelity

MidFi Mid-Fidelity



Abkürzungsverzeichnis

NUI Natural User Interface

OSC Open Sound Control

OTP One-Time-Pad

PIN Persönliche Identifikationsnummer

PKI Public Key Infrastructure

PKW Personen Kraftwagen

PRT Phrase Render Tool

SVG Scalable Vector Graphics

TAN Transaction Authentication Number

TLX Task-Load-Index

TTS Text-To-Speech

UDP User Datagram Protocol

UI User Interface

UX User Experience

VPC Value Proposition Canvas

VPN Virtual Private Network

VUI Voice User Interface

WIMP Windows, Icons, Mouse, Pointer

WOz Wizard-of-Oz



Abbildungsverzeichnis

1.1	Logo der adorsys GmbH & Co.KG [ado]	3
2.1	Infotainment-System eines Chevrolet mit Payment-Integration für Shell-Tankstellen (Quelle: [Pet18])	5
2.2	Brute-Force-Attacke bei alphanumerischen Passwörtern (Quelle: [Pas20])	10
2.3	Dhamija und Perrig (Quelle: [DP00])	11
2.4	Sobrado und Birget Triangle-Scheme (Quelle: [SB02])	11
2.5	Drawmetric-Systeme, links: DAS und rechts: Pass-Go (Quelle: [TA08])	12
2.6	Hot-Spots eines Beispiefotos (Quelle: [RDA09])	13
2.7	Einrichten des <i>Picture Password</i> (Quelle: [How17], [hel14])	14
2.8	Attacken auf grafische Passwörter (eigene Darstellung in Anlehnung an (Quelle: (vgl. [KAX11])))	15
2.9	Biometric-Glasses (Quelle: [SBBR16])	17
2.10	FFR und FAR biometrischer Systeme (Quelle: [HK17], [Fuj])	17
2.11	Klassische Aufteilung des Armaturenbrettbereichs (eigene Darstellung in Anlehnung an) (Quelle:[BBDV15, S. 15])	19
3.1	Value Proposition Canvas (eigene Darstellung in Anlehnung an [OPBS15])	24
3.2	Auswahl der Position für den Picture Passcode (Quelle-Hintergrund: [New09])	29
3.3	Markante Positionen mit Zahl (Quelle-Hintergrund: [New09])	30
3.4	Platzierung des Zahlenrasters (Quelle-Hintergrund: [New09])	31
3.5	Collagen mit Briefmarken und Wahrzeichen (Quelle:[Del15] [Bri]) . .	32
3.6	Icon Set und Wechsel der Zahl im Sekundentakt	33
3.7	Zusammengefügter Passcode	33
3.8	Picture Passcode Icon-Sets und Auswahl der Secrets	34
3.9	Anzeige zur Passcode Eingabe und erfolgreiche Autorisierung	35
3.10	Ähnlichkeiten unterschiedlicher Zahlen (Quelle: [Mis14])	38
3.11	Biometrisches Authentifizierungsverfahren iOS (Quelle: [Off18]) . . .	39



Abbildungsverzeichnis

4.1	Grafische Oberfläche der Fahrsimulation in Anlehnung an (Quelle:[Koh05])	42
4.2	Hauptpatcher der Fahrsimulation in Anlehnung an (Quelle:[Fod18]) .	44
4.3	Darstellung der OSC-Übertragung	46
4.4	Links: Operator-Panel, Rechts: Jeweiliger Tester-Panel	50
4.5	Testaufbau der Usability-Test	54
4.6	Messwerte der Usability-Tests	56
4.7	Ergebnisse der Nasa TLX des Usability-Tests (eigene Darstellung) .	57
4.8	Funktionsweise den Prototypen für den Picture Passcode	59
A.1	Ergebnisse der Pre-Tests	viii
A.2	Ergebnisse der offiziellen Usability-Tests	ix



1 Einleitung

In der folgenden Arbeit werden die Begriffe Picture Passcode, und Passcode gleichwertig verwendet. Es werden außerdem die Begriffe Benutzer und das englische Pendant User, Fahrer und Fahrzeugführer gleichgestellt benutzt. Die gewählte männliche Form bezieht sich immer zugleich auf weibliche und männliche Personen und fungiert einzig der besseren Lesbarkeit.

1.1 Ausgangssituation

Das Zusammenwachsen des Wirtschaftsraums EU bedingt, dass auch der Alltag der Menschen grenzübergreifend stattfindet. Es sind jedoch nicht die Ländergrenzen, welche den Bewegungsfluss immer wieder unterbrechen, sondern die unterschiedlichen Bezahlsysteme für Straßennutzungsgebühren der einzelnen Länder. Assistenzsysteme könnten hierbei die Handhabung für den Fahrer vereinheitlichen. Darüber hinaus haben sprachgesteuerte Systeme einen hohen Reifegrad erlangen können, weshalb das Bezahlen während der Fahrt via Sprache mittlerweile im Rahmen des Möglichen liegt [Ber18].

1.2 Zielsetzung

Bei einem sprachgesteuerten Bezahlsystem, stellt sich die Frage einer gesicherten Autorisierung. Wenn sich eine weitere Person im Fahrzeug befindet, muss die Sicherheit der Autorisierungsdaten dezidiert betrachtet werden. Das Ziel dieser Bachelorarbeit ist ein Interaktionskonzept auf Basis eines visuellen Passworts, welches als Picture Passcode bezeichnet wird, zu entwickeln. Um das entsprechende Verfahren nutzen zu können, müssen bestimmte Kriterien berücksichtigt werden. Dabei ist zunächst die Bedienbarkeit der Anwendung zu beachten. Dieses Kriterium ist dann erfüllt, wenn



1 Einleitung

der Nutzer die Anwendung während dem Fahren nutzen kann, ohne sich und andere im Straßenverkehr zu gefährden. Die zweite Voraussetzung ist die Sicherheit bezogen auf die Kombinatorik des Picture Passcodes. Hierbei sollen Autorisierungsdaten nicht enthüllt werden um den Zugriff auf das Konto des Nutzers zu schützen.

1.3 Vorgehensweise

Mittels eines selbstgebauten Prototypen wurde in Usability Tests untersucht, ob sich ein Picture Passcode als Autorisierungsverfahren eignet. Das Augenmerk lag auf der Bedienbarkeit und Sicherheit des Passcodes. Zusätzlich wurde eine Simulation aufgesetzt, welche die mentale, physische und zeitliche Beanspruchung des Fahrens simuliert um repräsentative Ergebnisse zu erhalten. Außerdem wurde die Anpassungsfähigkeit des Picture Passcodes an das Display unter Gewährleistung der Sicherheit betrachtet. Zur Unterstützung wurden einige Komponenten prototypisch umgesetzt.

1.4 Projektträger

Diese Arbeit wurde von der adorys GmbH & Co.KG unterstützt und von Steffen Blümm, dem Technical Lead des CUI-Bereichs, betreut. Im folgenden Abschnitt wird die adorsys in kurzen Worten vorgestellt. Die adorsys GmbH & Co. KG ist ein mittelständisches IT-Unternehmen, welches 2006 von Francis Pouatcha Nouyeuwe in Nürnberg gegründet wurde. Neben dem Gründer leiten Dr. Andrew J. Zeller und Stefan Hamm, aktuell das Unternehmen (Stand 26.05.2020). Es arbeiten an den Standorten Nürnberg und Frankfurt am Main derzeit 133 Mitarbeiter in einzelnen Projektteams, um den Kunden von der ersten Idee an bis zum fertigen Produkt zu begleiten.

Die adorsys bietet Software-Dienstleistungen im Bereich Digitalisierung für Unternehmen verschiedener Branchen mit besonderem Fokus auf Finanzdienstleistungen (insbesondere Banken und Versicherungen) an. Im Vordergrund stehen technologische Innovation, digitale Plattformen, Openbanking, Automatisierung, Mobility und Digitalisierung.



1 Einleitung

Zu den Kunden der Firma zählen unter anderem die TeamBank, die DATEV, die Bausparkasse Schwäbisch Hall oder die Senacor. Die adorsys GmbH & Co. KG schafft echte Wettbewerbsvorteile und hat aus diesem Grund in den letzten Jahren im Rahmen von externen Abschlussarbeiten und internen Projekten auch Kompetenzen im Bereich CUI aufbauen können [ado].



Abbildung 1.1: Logo der adorsys GmbH & Co.KG [ado]



2 Voraussetzungen zum Bezahlung während dem Fahren

Das folgende Kapitel beschreibt die benötigten Vorkenntnisse und Grundlagen die es essentiell sind, um eine Autorisierung für das Bezahlung während dem Fahren zu ermöglichen. Hierfür werden bisherige In-Car-Payment-Systeme untersucht und verschiedene Interfaces betrachtet, welche als mögliche Schnittstellen zwischen Fahrer und Service fungieren können. Als nächster Punkt werden allgemeine Authentifizierungs-/Autorsierungsverfahren unter sicherheitstechnischen Aspekten analysiert und für das Auto ausgelegt. Zuletzt wird die Ergonomie zur möglichen Verwendung des Passcodes im Fahrzeug charakterisiert.

2.1 Bisherige Systeme

In einer schnelllebigen und vernetzten Welt erwarten Menschen, dass sie jederzeit und überall Zahlungen veranlassen können. Der Aufenthalt in einem Auto stellt keine Ausnahme dar. Dazu ist eine Schnittstelle erforderlich, welche die Initiierung des Zahlungsvorgangs unterstützt, ohne die Hauptaufgabe des Fahrens zu beeinträchtigen. Aus diesem Grund haben einige Unternehmen der Finanz- und Automobilbranche unterschiedliche Systeme implementiert, die diesem Bedarf gerecht werden. Chevrolet beispielsweise hat ein integriertes In-Car-Zahlungssystem eingeführt, welches ermöglicht die Bezahlung des Fahrkraftstoffes einer Shell-Tankstelle sowie die Vorbestellung von Essen oder die Reservierung eines Restaurants über das Infotainmentsystem vom Auto aus zu erledigen (vgl. [Haw18]). Darüber hinaus bietet Honda, durch eine Kooperation mit Visa, die Möglichkeit mit seiner Visa Karte zu bezahlen. Dabei kann der Fahrer Bezahlungen initiieren, wenn das Visa-Symbol auf dem Touchscreen erscheint und durch Berühren eines Knopfes im Armaturenbrett betätigt wird. Exemplarisch kann damit unterwegs eine Essensbestellung zur Abholung erledigt werden (vgl. [Kor19]).

2 Voraussetzungen zum Bezahlung während dem Fahren



Abbildung 2.1: Infotainment-System eines Chevrolet mit Payment-Integration für Shell-Tankstellen (Quelle: [Pet18])

Die Firma Amazon bietet mit dem Sprachassistenten Alexa, einen Service an, bei dem es möglich ist eine Rechnung an jeder ExxonMobil und Exxon Tankstelle bezahlen zu können, ohne zur Kasse gehen zu müssen [Per20]. Zwar sind nicht alle Funktionen in Deutschland verfügbar, allerdings wurde ein System von Shell in Kooperation mit PayPal vorgestellt, bei dem es möglich ist das Benzin über das Smartphone zu bezahlen. Dabei muss sich der User eine App herunterladen, welche sich mit dem PayPal-Konto verbindet. Die App lokalsiert an welcher jeweiligen Shell-Station sich der Kunde aufhält und ermöglicht nach sicherheitstechnischen Abläufen den Bezahlvorgang (vgl. [NT18]). Daraus lässt sich ableiten, dass ein großes Interesse an solchen Systemen besteht, weshalb Mastercard auch auf der Suche nach Start-Ups ist, die eine sprachbasierte Zahlungslösung während dem Fahren anbieten [Mas20]. Ein prägnanter Nachteil der genannten Systeme ist jedoch der fehlende Autorisierungsschritt um eine Zahlung zu veranlassen. Da zu Beginn ein authentifiziertes Konto im System hinterlegt wurde, muss keine Abfrage zur Autorisierung initiiert werden. Falls sich der Fahrer jedoch nicht im Auto befindet, kann eine nicht autorisierte Person Zugriff auf das Konto erhalten um Zahlungen zu tätigen. Allerdings ist anzunehmen, dass insbesondere in Deutschland viele Menschen im Schutz ihrer Daten einen hohen Wert sehen. Aus diesem Grund bedingt es ein Autorisierungsverfahren, dass während der Fahrt genutzt werden kann und die Daten des Benutzers schützt.



2.2 Interfaces

In 2.1 wurde erläutert, dass bereits viele mögliche Bezahlssysteme im Auto existieren, welche über unterschiedliche Modalitäten gesteuert werden können. Es existiert die Möglichkeit, eine Zahlung über ein *Graphical User Interface* (GUI), wie beispielsweise einem Smartphone oder dem Display eines Infotainmentssystems, zu tätigen. Allerdings lässt sich eine Zahlung auch über ein *Voice User Interface* (VUI) initiieren.

Graphical User Interface

Bei der Verwendung des Begriffs GUI kann von zwei Dimensionen gesprochen werden. Es wird zwischen einem Interaktionsparadigma und Komponente unterschieden. Der technischer Layer in der Anwendung gilt als Komponente und dient als eine Benutzeroberfläche, um grafische Elemente des Systems darzustellen [Gra]. Auf der anderen Seite existiert das GUI als Interaktionsparadigma (WIMP). Das WIMP steht hierbei für *Windows, Icons, Menu and Pointer* und beschreibt die Modalität der Mensch-Maschinen-Schnittstelle [RA17]. Als Beispiel kann die Maus eines Standartrechners genannt werden, die in diesem Kontext als Pointer dient. Ein alternatives Interaktionsparadigma, auch *Native User Interface* (NUI) genannt, versucht dem User eine natürliche Interaktion mit dem System zu ermöglichen. Unter einem NUI zählt auch die Bedienung eines Smartphones oder einer berührungssensitiven Oberfläche [WW11, S. 12]. In seiner Dimension als technischer Layer besteht das GUI aus grafischen Bedienelementen, die nach dem Prinzip des *Recognition-over-Recall* aufgebaut sind, um einen einprägsamen Bedienvorgang zu ermöglichen.

Mit der Entwicklung des Automobils, wurde die Überlegung gemacht, Funktionalitäten des Fahrzeuges über einen Feedback-Loop sichtbar zu machen, um einen Überblick der Subsysteme für den Fahrer zu schaffen. Vor 30 Jahren wurde für die Steuerung der Klimaanlage, Radiolautstärke oder die Erhaltung von Informationen über das Auto, das erste GUI als technischer Layer implementiert [Hin16]. Diese Art wird als Infotainmentsystem bezeichnet, welches zentrale Bedienfunktionen des Autos zusammenführt, in einem Interface darstellt und über manuellem Input steuerbar gemacht wird. Dieses wird grundsätzlich als Display in der Mittelkonsole, Dashboard-Display oder als Head-Up-Display im Fahrzeug verbaut. Die Verwendung solcher Systeme dient



2 Voraussetzungen zum Bezahlen während dem Fahren

meist zur Navigation, Telefonie oder Musikauswahl. Die Bedienung eines solchen Systems muss nicht ausschließlich über integrierte GUI des Infotainmentsystems gesteuert werden. Mittlerweile besteht die Option, das Smartphone über Bluetooth mit dem Auto zu verbinden und einige Funktionalitäten zu steuern, wie in 2.1 beschrieben.

Voice User Interface

Neben einem GUI existiert in bestimmten Fahrzeugen die Möglichkeit, das Infotainmentsystem auch über Sprache ansteuern zu können. Ein solches Interface ist auch unter dem Begriff *Voice User Interface* (VUI) bekannt, bei dem die Interaktion über Sprache stattfindet. Verschiedene Unternehmen wie Amazon mit Alexa oder Apple mit Siri haben gezeigt, dass es bereits möglich ist Aufgaben über einen Sprachassistenten zu erledigen. Nachrichten, Erinnerungen oder Bezahlungen können somit auch während Situationen erledigt werden in denen ein eingeschränkter Blickkontakt herrscht oder eine freie Hand nicht zur Verfügung steht. Sprachassistentensysteme finden auch im Bereich der Automobilindustrie zunehmend Verbreitung. Laut einer Studie [InC20] haben knapp 130 Millionen Einwohner der Vereinigten Staaten von Amerika einen Sprachassistenten in einem Fahrzeug verwendet, 84 Millionen davon nutzen diesen regelmäßig und die Zahlen steigen kontinuierlich. Etwa ein Drittel der Sprachassistenten existieren als eingebettete Systeme im Auto. Die anderen zwei Drittel werden von externen Unternehmen wie Apple mit Siri oder CarPlay, durch die Verwendung des Smartphones über Bluetooth, bereitgestellt. Diese werden vorwiegend für den Beginn eines Telefonates, zur Navigation oder zum Verfassen einer Textnachricht, beansprucht [InC20, S. 6-15]. Der Grund weshalb Sprachassistentensysteme in Autos immer mehr Bedeutung erhalten ist die Verringerung der Ablenkungswirkung auf den Fahrer. Eine Studie [KV11, S. 162] hat hierbei bewiesen, dass der Fahrer bei einer manuellen Bedienung über ein Touch-Interface, 30 - 40 Prozent öfter den Blick abseits der Fahrspur hält, als bei einem sprachbasierten System. Zu beachten ist allerdings, dass alle Schritte auditiv nachvollziehbar sein müssen, damit der gewünschte Effekt eintreten kann.

Bei ansteigendem Task-Load kann es jedoch hilfreich sein, eine grafische Unterstützung, ergänzend zur mündlichen Konversation, zu erhalten. Eine mögliche Lösung hierfür bietet das *multimodale Conversational User Interface*. Diese Bezeichnung setzt sich aus den zwei Begriffen *multimodal* und *Conversational User Interface* (CUI) zusammen. Ein *Conversational User Interface* gilt als eine Benutzerschnittstelle, die den Ansatz der Konversationsführung zwischen Nutzer und System bildet.



2 Voraussetzungen zum Bezahlung während dem Fahren

Im genannten Use-Case, wären die Eingaben somit über die Sprache via eines Sprachassistentenzsystems erzielbar. Dies kann zunächst über einen *unimodalen* oder auch *multimodalen* Weg stattfinden. Eine *unimodale* Interaktion zwischen System und User findet ausschließlich über einen Kommunikationskanal statt. Als Beispiel gilt ein *voice-only* Gerät, welches die Sprache, als einziges Mittel zur Kommunikation mit dem System für den User vorgibt. Darunter zählen Smart-Speaker wie beispielsweise Google Home oder Amazon Echo [Coa18]. Bei einer *multimodalen* Kommunikation hat der User die Möglichkeit zur Verwendung mehrerer Modalitäten für den Informationsaustausch mit dem System (vgl. [Das18, S. 82]).

2.3 Authentifizierung und Autorisierung

Um eine Bezahlung ausführen zu können ist es notwendig, dass der User ein Konto bei einem System hinterlegt hat, damit dieses ausstehende Beträge abbuchen und ausgleichen kann. Für das Zahlen im Auto, ist das Konto mit dem integrierten System im Auto oder dem Smartphone, wie in Abschnitt 2.1 verbunden. Die Authentifizierung bzw. Autorisierung selbst kann hierbei über verschiedene Wege erfolgen, die im weiteren Verlauf des Textes beschrieben werden.

2.3.1 Wissensbasierte Verfahren

Ein wissensbasiertes Verfahren bildet hierbei das Passwort. Grundsätzlich dient ein Passwort zur Bestätigung der Authentizität der eigenen Person, sowie der Autorisierung zum Nachweis bestimmter Berechtigungen. Damit die Authentifizierung auch gewährleistet werden kann, darf das Passwort keiner dritten Person bekannt sein [Pas]. Die gezielte Unterscheidung zwischen textuell - und grafischen basierten Passwörtern ist für das Folgende essentiell.



2 Voraussetzungen zum Bezahlend während dem Fahren

2.3.1.1 Persönliche Identifikationsnummer

Eine persönliche Identifikationsnummer (PIN) ist eine spezielle Form des Passwortes. Der Unterschied zwischen einer PIN und einem Passwort liegt darin, dass sie als eine personenbezogene Nummer gilt. Diese wird in den meisten Fällen, in Kombination eines physischen Objektes wie einer EC-Karte, eines Smartphones oder Computer in Verbindung gebracht und als zweiter Sicherheitsfaktor verwendet, um sich gegenüber Services als rechtmäßiger Besitzer des Objektes zu authentifizieren (vgl. [PIN]). Eine PIN besteht meist aus einer Folge von vier bis acht Zeichen, jedoch hat sich die vierstellige Ziffernfolge durchgesetzt [Mil07]. Eine vierstellige Ziffernfolge bietet $10^4 = 10.000$ Kombinationsmöglichkeiten. Diese Zahl scheint zunächst sehr niedrig, allerdings sind durchweg zusätzliche Sicherheitsaspekte zu beachten, die dieses Defizit ausgleichen. Eine PIN erfordert in den meisten Fällen eine manuelle Eingabe, was eine Brute-Force-Attacke kaum möglich macht, da diese viel Zeit in Anspruch nehmen würde. Zusätzlich legen viele Systeme eine maximale Anzahl an Versuchen für die Eingabe der PIN fest. Falls das Maximum der Eingabe überschritten wurde, verweigert das System im Anschluss den Zugriff auf das Konto und eine Authentifizierung ist zunächst nicht möglich. Ein Angreifer hätte bei einer Attacke mit vier Versuchen somit nur eine Wahrscheinlichkeit von 1/2500 für das Erraten des Passwortes (vgl. [PIN17]).

2.3.1.2 Alphanumerische Passwörter

Alphanumerische Passwörter werden oft zur Authentifizierung und oder zur Datensicherheit verwendet. Um eine vernünftige Maß an Sicherheit zu erreichen, wird oft die Verwendung einer bestimmten Anzahl an Sonderzeichen, Nummern oder Buchstaben vorgeschrieben. Problematisch erweist sich die, durch die Komplexität des Passworts, erschwerte Einprägsamkeit für den Benutzer. Dies hat zur Folge, dass die Mehrzahl ein kurzes, simples Passwort mit persönlichen Hintergrund wählt [Cen10, S. 3]. Dies stellt ein erhöhtes Sicherheitsrisiko dar, da diese leicht zu ermitteln sind und oft auf vielen Services gleichzeitig verwendet werden. Insbesondere bei Services, bei welchen automatisierte Attacken möglich sind, erweist sich dies als hohes Sicherheitsrisiko. Angriffe können beispielsweise über eine Brute-Force-Attacke, wie in 2.2 zu sehen, initiiert werden (vgl. [BSIb]).

2 Voraussetzungen zum Bezahlend während dem Fahren

Zeichenraum	Maximale Rechenzeit eines Brute-Force-Angriffs bei 1 Milliarde Schlüsseln pro Sekunde									
	Passwortlänge									
	4 Zeichen	5 Zeichen	6 Zeichen	7 Zeichen	8 Zeichen	9 Zeichen	10 Zeichen	11 Zeichen	12 Zeichen	
10 [0–9]	<1 ms	<1 ms	1 ms	10 ms	100 ms	1 Sekunde	10 Sekunden	2 Minuten	17 Minuten	
26 [a–z]	<1 Sekunde	<1 Sekunde	<1 Sekunde	8 Sekunden	4 Minuten	2 Stunden	2 Tage	42 Tage	3 Jahre	
52 [A–Z; a–z]	<1 Sekunde	<1 Sekunde	20 Sekunden	17 Minuten	15 Stunden	33 Tage	5 Jahre	238 Jahre	12.400 Jahre	
62 [A–Z; a–z; 0–9]	<1 Sekunde	<1 Sekunde	58 Sekunden	1 Stunde	3 Tage	159 Tage	27 Jahre	1.649 Jahre	102.000 Jahre	
96 (+Sonderzeichen)	<1 Sekunde	8 Sekunden	13 Minuten	21 Stunden	84 Tage	22 Jahre	2.108 Jahre	202.000 Jahre	19 Mio. Jahre	

Abbildung 2.2: Brute-Force-Attacke bei alphanumerischen Passwörtern (Quelle: [Pas20])

2.3.1.3 Grafische Passwörter

Grafische Passwörter kamen zum ersten mal 1996 durch den Wissenschaftler Blonder zur Sprache [Blo96]. Ein grafisches Passwort dient wie die vorherigen Methoden, als eine Authentifizierungs- bzw. Autorisierungsmethode. Der Unterschied besteht hier allerdings darin, dass sich der User sein Passwort aus verschiedenen grafischen Elementen zusammenstellt (vgl. [Rou07]). Der Grund, weshalb grafische Passwörter als Methode zum Schutz von sensiblen Daten verwendet werden, liegt bei dem *Picture-superiority-effect* [NRW76]. Dieser beschreibt den Effekt, dass sich der Mensch grafische Darstellungen besser einprägen kann als alphanumerische Elemente. Ein weiterer Grund grafische Passwörter zu verwenden, ist die höhere Sicherheit in Bezug auf externe Angriffe. Eine *Dictionary-Attacke* oder das Auffangen des Passwortes eines klassischen *Keylogger* über eine Tastatur sind nicht möglich [GW16]. Die Methoden grafischer Passwörter werden hierbei in *Recognition-, Recall- und Cued-Recall-based* Verfahren unterteilt.

Recognition-based Verfahren

Die *Recognition-based*-Methode verfolgt den Ansatz, dass der User, zuvor ausgewählte grafische Elemente, aus einem Set anderer Elemente unterscheiden, erkennen und in der richtigen Reihenfolge identifizieren kann (vgl. [GW16, S. 492]). Dhamija und Perrig haben beispielsweise ein grafisches Autorisierungsverfahren entwickelt, indem ein Programm ein Raster an randomisierten Bildern erstellt. Der User muss sich ein Set aus verschiedenen Bildern zusammenstellen und diese im Autorisierungsprozess in der richtigen Reihenfolge wiedergeben in welcher er die Bilder ausgesucht hat [DP00].

2 Voraussetzungen zum Bezahlung während dem Fahren

Da ein Passwort aus fünf Elementen besteht und das Bilderset aus 15 Elementen, beträgt die Wahrscheinlichkeit die richtigen Bilder und deren Reihenfolge zu erraten, hier bei $1/\binom{20}{5} = 1/15504$.

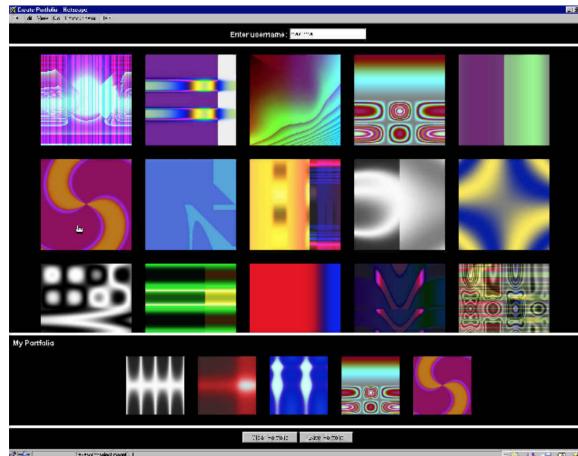


Abbildung 2.3: Dhamija und Perrig (Quelle: [DP00])

Ein ähnliches Verfahren von Sobrado and Birget, siehe 2.4 [SB02], löst das Problem des *Shoulder-Surfs*. Dies bedeutet, dass eine andere anwesende Person nicht in der Lage ist das Passwort, durch bloßes Beobachten, zu entschlüsseln (vgl. [Sho]).

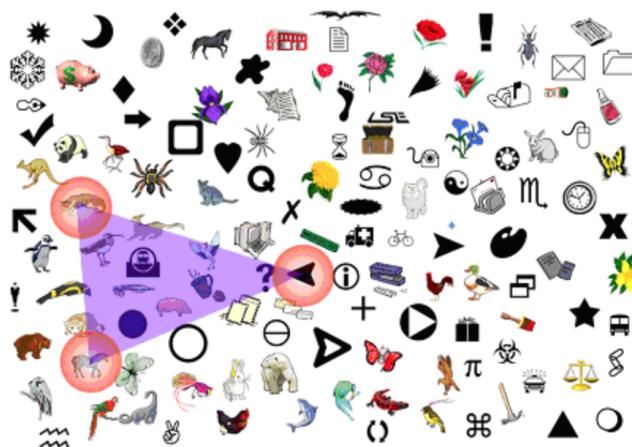


Abbildung 2.4: Sobrado und Birget Triangle-Scheme (Quelle: [SB02])

Das System verteilt eine Anzahl an verschiedenen Objekten auf einer Oberfläche. Diese ähneln sich zunächst, sind bei genauerer Betrachtung allerdings unterscheidbar. Daraufhin muss sich der User eine Menge aus den angezeigten Objekten heraussuchen und merken. Hierbei spielt die Reihenfolge der Bilderauswahl keine Rolle. Für den Autorisierungsprozess, streut das System wieder unterschiedliche Objekte auf die gra-

2 Voraussetzungen zum Bezahlung während dem Fahren

fische Benutzeroberfläche. Das System wählt nun zufällig einen Bereich innerhalb der Hälfte des Bildschirms aus und platziert die ausgewählten Objekte darin. Diese bilden nun ein unsichtbares Dreieck. Um sich anzumelden, muss der Benutzer in das imaginäre Dreieck klicken. Die Kombinationsmöglichkeiten des Passwortes, liegen hier bei $\binom{1000}{10} \approx 2.6 * 10^{23}$, wenn das Set aus $N = 1000$ Elementen und $K = 10$ ausgewählten Elementen besteht (vgl. [SB02]).

Recall-based Verfahren

Die *Recall-based*-Methode beschreibt das Konzept der Reproduktion oder Abrufen einer Grafik, die in der Registrationsphase erstellt oder ausgewählt wurde (vgl. [GW16]). Diese Methode lässt sich wiederum in zwei Kategorien unterteilen [SB13].

Das beste Beispiel für eine *Pure-Recall-based*-Technik liefert das *Draw-a-Secret* (DAS) Verfahren [JMM⁺99]. Um das System nutzen zu können, muss der User eine Zeichnung auf einem 2D-Raster mit einer Größe von $G \times G$, zeichnen. Er muss darauf achten in welchen Rasterflächen und in welcher Malabfolge er das Bild gezeichnet hat. Dies ist ein wichtiger Aspekt, da das System das vorgezeichnete Passwort, mit dem gespeichertem Passwort mittels der obengenannten Kriterien vergleicht. Der Vorteil solcher Systeme ist, dass keine grafische Datenbank zur Nutzung gebraucht wird, da die gemalten Punkte als Zahlenwerte gespeichert werden. Da es bei jedem Zeichnen eine gewisse Standardabweichung zum hinterlegtem DAS-Passwort gibt, werden *Sensitive-Areas* hinzugefügt, die einen bestimmten Spielraum zu Abweichungen geben [JMM⁺99].

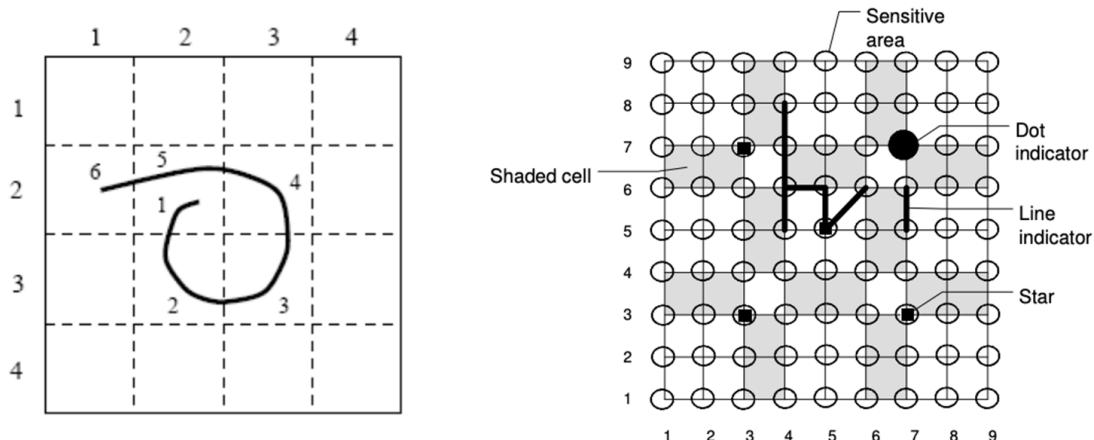


Abbildung 2.5: Drawmetric-Systeme, links: DAS und rechts: Pass-Go (Quelle: [TA08])

2 Voraussetzungen zum Bezahlend während dem Fahren

Bei der *Cued-Recall-based*-Methode wird dem Benutzer zur Authentifizierung ein Hinweis zum Abrufen eines Kennworts bereitgestellt, der während der Registrierungsphase erstellt wurde (vgl. [GW16, S. 493]). Als Beispiel kann die *Pass-Point*-Methodik angegeben werden, bei dem der User gebeten wird, das Passwort durch klicken auf bestimmte Bereiche eines pixel-basierten Fotos, zu bestätigen. Im Detail soll er Klickpunkte, auch *Pass-Points* genannt, auf einem Foto, mit einer geordneten Folge abarbeiten. In diesem Fall hat der User einen bestimmten Toleranzbereich, um die jeweiligen Punkte zu treffen. Im Vergleich zur *Pure-Recall-based*-Methode, wird ein Foto als eine Orientierungshilfe verwendet, damit sich der User einfacher an die Reihenfolge und Position der *Pass-Points* erinnert. Dabei entsteht das Phänomen der *Hot-Spots*. Dies sind bestimmte Bereiche eines Bildes, die häufiger geklickt werden als andere. Dies hat zur Folge, dass gewisse Positionen mit einer prägnanten Lage schnell zu erraten sind. Zudem kann ein Wörterbuch aus häufig geklickten Punkten gebildet werden, welche durch Bildverarbeitung ermittelt werden können. Des Weiteren hat sich herausgestellt, dass viele Punkte auch einem Muster nachgehen können (vgl. [CFBv09]).



Abbildung 2.6: Hot-Spots eines Beispielfotos (Quelle: [RDA09])

Blackberry Picture Password

Blackberry hat 2014 mit dem BlackBerry OS 10.2.1 Update das *Picture Password* auf dem Markt eingeführt. Dadurch wurde eine weitere Möglichkeit geboten das Smartphone nicht nur über eine PIN oder Zeichenmuster, sondern auch über ein grafisches Passwort, zu entsperren [hel14]. Das *Picture Password* verfolgt den *Cued-Recall-based*-Ansatz.

2 Voraussetzungen zum Bezahlung während dem Fahren

Funktionsweise des Picture Password

Zunächst wählt der Benutzer ein Foto aus, welches er für das Picture Password verwenden möchte. Dieses kann er aus einer Reihe von Fotos, welche vom System vorgeschlagen werden, aussuchen oder der Nutzer verwendet ein eigenes. Es ist allerdings zu empfehlen ein Bild zu bestimmen, welches viele *Hot-Spots* aufweist. Im Anschluss selektiert der User eine bestimmte einstellige Zahl aus einem Zahlenspektrum von 0-9.

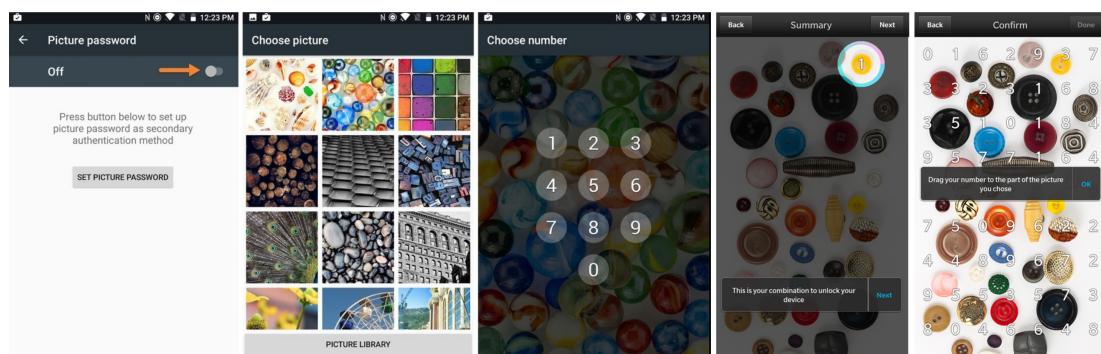


Abbildung 2.7: Einrichten des *Picture Password* (Quelle: [How17], [hel14])

Beim Login erscheint die gewählte Nummer auf dem zuvor gewähltem Bild, die er an eine gewünschte Stelle im Bild positioniert. Diese Zahl ist zudem in einem Raster mit unterschiedlichen einstelligen Zahlen eingebettet. Um sich nun einzuloggen muss der User die Zahl an die zuvor gewünschte Position verschieben (vgl. [hel14]).

Das Picture Password ist im Vergleich zu einem klassischen vierstelligen PIN, nicht anfällig für *Shoulder-Surfing*. Da der Benutzer nicht nur die Zahl sondern das gesamte Zahlenraster verschiebt, ist es kaum möglich zu ermitteln, welche bestimmte Zahl an welche bestimmte Stelle verschoben wird. Der Grund dafür ist, dass für jedes neue Einloggen, ein Zahlenraster generiert wird, welches mit unterschiedlichen Zahlen und Abständen zwischen den Zahlen versehen ist. Wäre dies nicht der Fall, könnten einige der Zahlen immer wieder auf gleiche Stellen bzw. Hot-Spots positioniert werden. Daraus folgt die Konsequenz, dass es angreifbar unter *Shoulder-Surfing* wäre (vgl. [How14]).

Angriffsszenarien grafischer Passwörter

Viele der vorgestellten Verfahren bieten zwar einen gewissen Sicherheitsfaktor, jedoch kann dieser Aspekt durch bestimmte Attacken umgangen werden.

2 Voraussetzungen zum Bezahlend während dem Fahren

Attacken auf grafische Passwort Verfahren						
Grafische Passwort Verfahren	Typ des Schemas	Brute Force Attack	Dictionary Attack	Spy-ware or Naive Key logging	Shoulder Surfing Attack	Phishing Attacke oder Social Engineering
BLONDER	Recognition Based	Y	N	N	Y	N
Dhamija & Perrig	Recognition Based	Y	X	X	X	X
Sobrado and Birget	Recognition Based	N	Y	X	Y	X
DAS	Pure Recall Based	N	Y	N	Y	N
PassGo	Pure Recall Based	Y	X	X	X	X
PassPoints	Cued Recall Based	Y	N	N	Y	N
Blackberry Picture Password	Cued Recall Based	Y	Y	Y	Y	Y
	Y = Ja ist resistent N = Nein, ist nicht resistent	N = Nein, ist nicht resistent	X = nicht erforscht			

Abbildung 2.8: Attacken auf grafische Passwörter (eigene Darstellung in Anlehnung an (Quelle: (vgl. [KAX11])))

In der 2.8 Abbildung ist zu sehen, welche Verfahren gegen bestimmte Attacken resistent sind. In diesem Vergleich schneidet hierbei das *Picture Password* von Blackberry am Besten ab, da es gegen elementare Attacken geschützt ist. Eine Brute-Force-Attacke kann nicht initiiert werden, da die Autorisierung zunächst nur lokal auf dem Smartphone stattfindet, weshalb ein automatisierter Angriff sehr unwahrscheinlich ist. Eine Dictionary-Attacke und Key-Logging sind nicht möglich, da der User keine textuellen Eingaben macht. Das Shoulder-Surfing wird wie im oberen Teil beschrieben über unterschiedliche Anordnung der Zahlen verhindert. Zuletzt lässt sich das Passwort nicht durch eine Phishing-Attacke oder Social-Engineering attackieren, da es keinen Spielraum für einen persönlichen Bezug bietet. Das *Picture Password* bildet somit ein solides Fundament als Ausgangspunkt.

2.3.2 Besitz

Unter Besitz kann die Verwendung eines physikalischen Objektes, wie einer EC-Karte oder einem Schlüssel, zur Authentifizierung der Person verstanden werden. Nur der alleinige Besitz eines Objektes reicht meist allerdings nicht aus, da es gestohlen, übertragen oder verloren gehen kann. Aus diesem Grund wird oft eine Zwei-Faktor-Authentisierung veranlasst, die nach einem zusätzlichen Faktor, wie einem wissensbasierten Verfahren verlangt (vgl. [war]). Einmalkennwörter (OTP) oder digitale Zertifikate fallen zudem unter den Begriff Besitztum. Einmalkennwörter werden ein einziges mal verwendet und bei jedem Login dynamisch neu generiert. Dieses Verfahren ist vergleichbar mit dem TAN-Verfahren beim Online-Banking, welches automatisch bei jedem Login oder Überweisung eine TAN generiert, um sich zu authentifizieren oder



2 Voraussetzungen zum Bezahlend während dem Fahren

autorisieren. Dabei wird meist zusätzlich ein physikalisches Besitztum verlangt. Ein digitales Zertifikat wiederum ist ein Nachweis welcher elektronisch von einer Zertifizierungsstelle ausgestellt wird. Mit Hilfe der Public-Key-Infrastruktur (PKI) ist es möglich Informationen verschlüsselt zu übertragen. Da ein Zertifikat, einen öffentlichen Schlüssel generiert, der einem bestimmten Besitzer zu gewiesen wird, kann dieses herausfiltern wem der öffentliche Schlüssel gehört. Dieses Verfahren wird meist dazu verwendet um E-Mails zu verschlüsseln, digitale Dokumente zu signieren oder VPN-Verbindungen aufzubauen (vgl. [For14]).

2.3.3 Biometrische Verfahren

"Biometrische Erkennung durch ein biometrisches System verfolgt das Ziel, eine mittels automatisierter Messung durch ein spezifisches Merkmal bestimmte Person von anderen unterscheidbar zu machen."(vgl. [BSIa]). In anderen Worten bedeutet das, dass sich eine Person durch bestimmte Körpermerkmale authentifizieren oder autorisieren kann. Dabei gibt es des Verfahren der Stimm- bzw. Sprecherkennung, Fingerabdruck-, Gesichts- oder der Iriserkennung. Die genannten Systeme testen hierbei nicht, ob eine bestimmte Person die genau gleichen Daten aufweist die im System gespeichert sind, sondern nur die hinreichende Ähnlichkeit der Person.

Hervorzuheben ist der Vorteil solcher Systeme, da biometrische Merkmale einer Person direkt zugeordnet sind und diese nicht von einer dritten Person auf natürlichem Wege simuliert werden können. Biometrische Merkmale sind nicht natürlich veränderbar, da markante Körpermerkmale, wie die Iris oder der Fingerabdruck, sich im Laufe der Jahre kaum verändern. Das birgt den Vorteil, dass das System prägnant lange verwendet werden kann (vgl.[BSIa]). Im Vergleich zu einem alphanumerischen Verfahren, weist ein biometrisches System eine inhärente Komplexität, um seine Daten zu schützen. Jedoch kann sich auch hier Zugriff zu den Daten verschaffen werden. Ein Team aus Wissenschaftlern der Carnegie Mellon University hat beweisen können, dass es machbar ist ein Gesichtserkennungssystem dazu zu bringen Gesichter falsch zu identifizieren oder als Person unkenntlich zu machen. Dies wurde durch eine spezielle Brille hervorgerufen, zu sehen in Abbildung 2.9, mit der es beispielsweise möglich war auch kommerzielle 2D-Gesichtserkennungssysteme wie *Face++* zu täuschen (vgl. [Ger16], [SBBR16]).

2 Voraussetzungen zum Bezahlen während dem Fahren

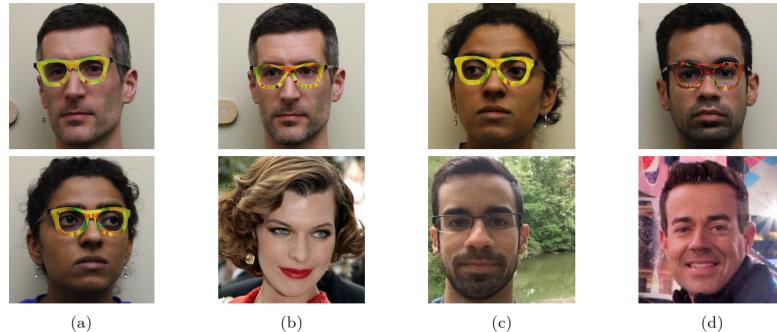


Abbildung 2.9: Biometric-Glasses (Quelle: [SBBR16])

Auch im Bereich der Stimm - und Sprecherkennung gibt es sicherheitstechnische Bedenken. Künstliche Intelligenzen wie Lyrebird [Lyr] oder Programme wie Adobe VoCo [VoC16] können durch ausreichende Trainingsdaten, Stimmen simulieren. Über die Texteingabe kann dann festgelegt werden, was die simulierte Stimme sagen soll. Das kann zu einem Missbrauch führen, um sich beispielsweise als eine andere Person auszugeben. Dies ist besonders kritisch im Bezug auf den Zugriff auf sensible Daten. Auf der anderen Seite, können Stimmen auch von einer anderen realen Personen imitiert oder die eigentliche Stimme kann durch eine Erkrankung verzerrt werden. Diese Bedenken werden von Google im Bezug auf ihr hauseigenes *Voice-Match-Verfahren* [Por20] bestätigt, weshalb ein Hinweis vor den Gefahren eines Missbrauchs warnt.

Falscherkennungsrate (FAR) und Falschrückweisungsrate (FRR)		
Authentifizierungsmethode	FAR (%)	FRR (%)
Gesichtserkennung	≈ 1.3	≈ 2.6
Sprachmuster	≈ 0.01	≈ 0.3
Fingerabdruck	≈ 0.001	≈ 0.1
Fingervene	≈ 0.0001	≈ 0.01
Iris/Retina	≈ 0.0001	≈ 0.01
Fujitsu Palm vein	< 0.00001	≈ 0.01

Abbildung 2.10: FFR und FAR biometrischer Systeme (Quelle: [HK17], [Fuj])

Die Fehlerraten solcher Systeme werden unter *False Rejection Rate* (FFR) und *False Acceptance Rate* (FAR) gemessen. "Die FFR ermittelt den prozentualen Anteil, welche Personen fälschlicherweise zurückgewiesen wurden. Die FAR gibt wiederum an welchen prozentualen Anteil der Personen fälschlicherweise zuordnet und zulässt." (vgl. [Int]).



Demzufolge ist die Sicherheit alleinig durch biometrischen Merkmalen nicht garantiert, weshalb es sich empfiehlt einen zusätzlichen Authentifizierungsfaktor hinzuzunehmen.

2.4 Ergonomie im Fahrzeug

Der Kontext für das Zahlen im Auto beinhaltet zwangsläufig die Benutzung eines Fahrzeuges. Die Fahraufgabe wird hierbei in drei Kategorien geteilt (vgl. [BVG15, S. 20ff]). Unter primäre Aufgabe des Fahrens wird das Navigieren, die Führung und Stabilisierung des Fahrzeuges verstanden. Die sekundäre Aufgabe beinhaltet wiederum die Tätigkeiten in Abhängigkeit von Fahranforderungen wie Aktionen und Reaktionen auf bestimmte Verkehrssituationen. Darunter zählt beispielsweise die Betätigung des Blinkers um eine Spur zu wechseln. Das Bezahlten während dem Fahren ist wiederum als teritäre Aufgabe anzusehen, da sie mit dem eigentlichen Fahren nichts zu tun hat. Eine der wichtigsten Maxime für den Einsatz von Technologien im Auto ist die Gewährleistung der Sicherheit im Straßenverkehr. Aus diesem Grund wird diese unter einigen Punkten der Verkehrspychologie, Wahrnehmungspsychologie und Aufmerksamkeit des Fahrers betrachtet.

2.4.1 Verkehrspychologie

Die Verkehrspychologie beschäftigt sich mit der Analyse und Forschung des menschlichen Verhaltens im Straßenverkehr (vgl.[DGV]). Hierbei wird der Use-Case für das Zahlen während dem Fahren über ein grafisches Autorisierungsverfahren betrachtet. Dieser beinhaltet zusätzlich zur Aufgabe des Fahrens, die Bewältigung einer teritären Aufgabe. Diese wird in der Regel über ein Fahrerinformationssystem (FIS) vollzogen, da sie nicht zur primären Fahraufgabe zählt. Mittlerweile ist es jedoch möglich, im Bereich des Fahrerassistenzsystems (FAS) drittrangige Fahraufgaben zu erledigen. Eine Nebenaufgabe welches über ein FIS oder FAS abgewickelt wird, kann zur visuellen Ablenkung, mentalen Beanspruchung oder zur biomechanischen Störung führen (vgl. [KV11, S. 170ff]).

Die visuelle Ablenkung wird verursacht, wenn bei der Nutzung eines Systems eine Blickabwendung, weg von der Fahrsituation benötigt wird.

2 Voraussetzungen zum Beziehen während dem Fahren

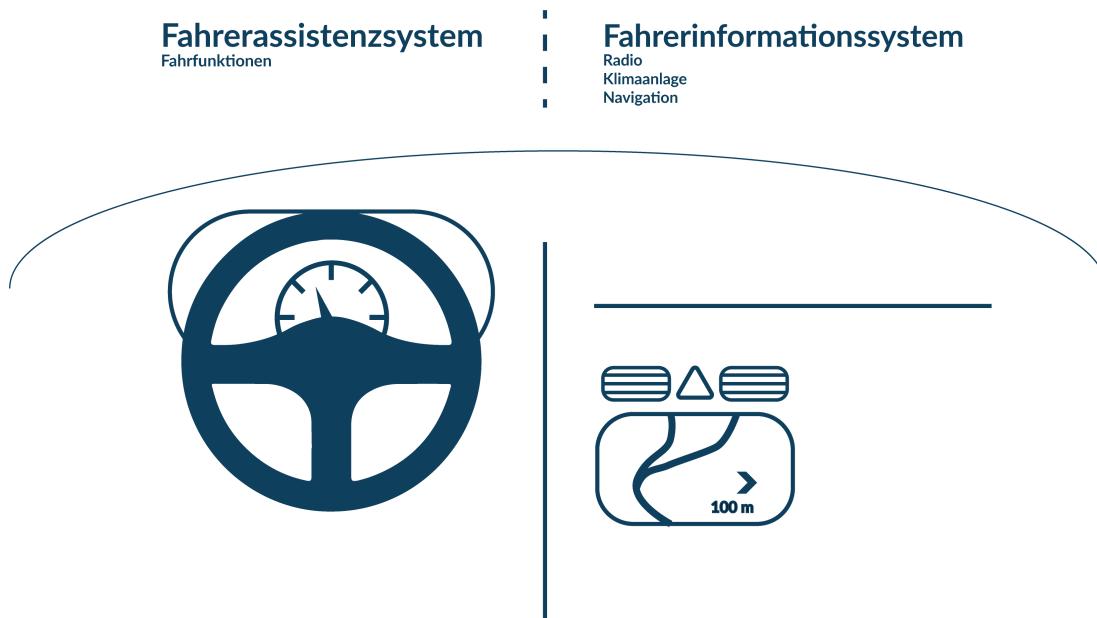


Abbildung 2.11: Klassische Aufteilung des Armaturenbrettbereichs (eigene Darstellung in Anlehnung an) (Quelle:[BBGV15, S. 15])

Dies führt dazu, dass in den meisten Fällen das Halten der Fahrspur beeinträchtigt und das Risiko eines Verkehrsunfalles erhöht wird (vgl. [KV11, S. 170ff]). Um ein solches System trotzdem bedienbar und verkehrstechnisch sicher zu gestalten, stellte die Europäische Union, 2000 (2008 überarbeitete Version) eine Empfehlung (*European statement of principles on human machine interface for in-vehicle Information and communication systems* [Com08]), zum Design von Interfaces innerhalb eines Fahrzeugs, aus. Diese soll zur Orientierung für die Entwicklung eines FIS-Designs dienen, damit das Unfallrisiko durch die Bedienung dessen nicht erhöht wird. Die Empfehlung besagt, dass die vom System dargestellte Information mit wenigen Blicken erfassbar sein und zu einem angemessenen Zeitpunkt gegeben werden soll. Das System darf keine langen und nicht-unterbrechbaren Befehlssequenzen bzw. Blickabwendungen erfordern. Zusätzlich muss das System eine ausreichende und konkrete Informationsdarstellung bieten, da es sonst zu häufigeren und längeren Blickabwendungen führen kann. Die Prinzipien wurden 2001 auch von der Vereinigung der europäischen Autohersteller (ACEA) angenommen, um sich bei der Gestaltung von Informationssystemen für Fahrer zu orientieren (vgl. [KV11, S. 173]). Hierbei kam zusätzlich der Punkt dazu, wie lange die Blickabwendungsdauer zur Fahrbahn betragen darf. Diese wurde auf zwei Sekunden festgelegt, damit der Fahrer sein Blickfeld weiterhin auf die Fahrbahn legen kann. Außerdem darf die Bearbeitung einer Aufgabe eine gesamte Blickabwendungsdauer von maximal 20 Sekunden benötigen, damit der Fahrzeugführer die Konzentra-



2 Voraussetzungen zum Bezahlend während dem Fahren

tion weiterhin auf die eigentliche Fahraufgabe legen kann (vgl. [Met09, S. 30]). Damit all diese Punkte erfüllt werden können, ist es somit essentiell, eine ausreichende und übersichtliche Darstellung von Informationen zu gewährleisten.

Allerdings hat muss auch beachtet werden, dass nicht nur die visuelle Ablenkung einen Einfluss auf das Fahrverhalten birgt, sondern auch die mentale Beanspruchung. Gespräche oder Telefonate können zu einer Störung führen, obwohl der Blick des Fahrers auf die Fahrbahn gerichtet bleiben kann. Das Verkehrsgeschehen wird dadurch weniger bewusst wahrgenommen und verarbeitet. Das Sehfeld verengt sich und das Blickverhalten wird einförmiger. Dies kann dazu führen, dass die Verkehrsregelungen nicht mehr verfolgt werden können und zur einer verzögerten Wahrnehmung von Gefahren resultiert. Folglich beeinträchtigt dies die Verkehrssicherheit (vgl. [KV11] S. 171), [HNE00]. Aus diesem Grund empfiehlt sich ein System zu gestalten, welches geringes Nachdenken erfordert und wenig komplexe Entscheidungen beinhaltet.

Die biomechanische Störung kann zu einer Beeinträchtigung des Fahrens führen. Darunter werden physische Aspekte, wie der Veränderung der Körperposition gezählt (vgl. [KV11, S. 171f]).

2.4.2 Wahrnehmungspsychologie

Die Wahrnehmung als einzelner Begriff ist der Prozess, Informationen durch Sinnesorgane aufzunehmen und zu verarbeiten (vgl. [Cog]). Wie der Mensch etwas wahrnimmt, hängt von biologischen (körperliche Leistungsfähigkeit), psychologischen (Gedächtnis, Aufmerksamkeit) und anderen Umwelteinflüssen (Wetter, Lichtverhältnisse) ab. Die benötigten relevanten Informationen, werden zu 90 Prozent visuell wahrgenommen und die anderen zehn Prozent über den Hör-, Gleichgewichts - und Tastsinn (vgl. [CRZ11, S. 59]). Eine visuelle Nebenaufgabe beansprucht im Mittel 1.2 Sekunden einer Blickabwendung (vgl. [CBW04]), [Met09, S. 22ff]. Dabei spielt nicht das Erkennen eines Objektes die größte Rolle, sondern die Dauer der Fixation. Das bedeutet, dass der Fahrer ein bestimmtes Objekt gezielt fixiert und betrachtet. Die Fixation der Straße beispielsweise benötigt nur die Hälfte der Zeit, im Vergleich zur Fixation einer Nebenaufgabe, wie der Blick auf das Navigationssystem. Aus diesem Grund sollten einzelne Blickabwendungen nicht länger als 1,2 Sekunden betragen, da der Fixationswert noch mit einberechnet werden muss [ZAD00].

Der Begriff Fixation kommt häufig zur Sprache, wenn unter fovealen und peripheren



2 Voraussetzungen zum Beziehen während dem Fahren

Sehen unterschieden wird. Beim fovealen Sehen hat der Mensch einen Bereich, indem er einen kleinen Teil des visuellen Wahrnehmungsfeldes, detailliert wahrnehmen kann. Das beträgt nur zwei Prozent des Gesichtsfeldes (vgl. [KV11, S. 29]).

Da das foveale Sehen durch zusätzliche Nebenaufgaben beeinträchtigt werden kann, dürfen maximal drei Blicke mit einer Blickdauer von 2,0 Sekunden benötigt werden, um die Nebenaufgabe zu erfüllen. Wenn mehr Blicke und Zeit verwendet wird, beginnt der sicherheitskritische Bereich (vgl. [MDH⁺07]). Beim peripheren Sehen ist es wiederum so, dass der daneben gelegene extra-foveale Bereich wahrgenommen wird. Beim Fahren kann durch peripheres Sehen somit der Defizit bei der Leistung der Spurhaltung ausgeglichen werden, wenn der Fahrer beispielsweise wegschaut. Zwar reicht peripheres Sehen zur Identifikation von Objekten nicht aus, allerdings können Veränderungen entdeckt werden, die zur Abwägung handlungsrelevanter Informationen durch Aufmerksamkeit führen. Das periphere Sehen wird durch kognitiv beanspruchende Aufgabe, wie Telefonate, beeinträchtigt. Dies führt dazu, dass die Aufnahmefähigkeit (Tunnel-Vision) sinkt und zu einem Risikofaktor im Straßenverkehr führt (vgl. [KV11, S. 29ff]).

2.4.3 Aufmerksamkeit

Die Aufmerksamkeit ist die gezielte Auswahl von Informationen, die zu Inhalten bewusster Wahrnehmung werden sollen. Dabei entscheidet jede Person selbst, worauf der Fokus gelegt wird, da das Wahrgenommene mit vergleichbar erlebten Situationen abgeglichen und abgeschätzt wird. Daraus resultiert, ob bestimmten Dingen Aufmerksamkeit geschenkt wird (vgl. [KV11, S. 28f]). Wenn diese Selektion nicht stattfinden würde, führt dies zu einer Reizüberflutung, da die visuelle Kapazität des Menschen übersteigt.

Dieses Phänomen wird auch als selektive Aufmerksamkeit bezeichnet und bedeutet, dass die Blickschärfe auf relevante Aspekte des Gesichtsfeldes verlagert wird. Dieser Prozess wird unbewusst gesteuert und, wie im oberen Teil beschrieben, wird nach einem Muster gesucht, um irrelevante Aspekte auszublenden. Die Besonderheiten, die bei einer selektiven Aufmerksamkeit vorkommen sind die *Inattentional-Blindness* und oder *Change-Blindness*. Die *Inattentional-Blindness* sagt aus, dass bestimmte Objekte nicht wahrgenommen werden, wenn keine Aufmerksamkeit auf sie gelegt wird.



2 Voraussetzungen zum Bezahlen während dem Fahren

Eine weitere Stufe ist die *Change-Blindness* in dem der Fahrer Veränderungen an anderen Objekten übersieht, da die Aufmerksamkeit auf etwas anderes gelegt wird (vgl. [CRZ11, S. 65]).

Zusätzlich zur selektiven Aufmerksamkeit existiert die geteilte Aufmerksamkeit. Hierbei werden zwei oder mehrere Aufgaben zur selben Zeit bearbeitet. Der Unterschied ist allerdings, dass eine Aufgabe automatisiert abläuft und keiner weiteren bewussten Aufmerksamkeit bedarf. Dieser Fall ist vergleichbar mit einem routinierten Fahrer, der mental nicht damit beschäftigt ist die Gangschaltung zu betätigen. Dadurch kann er sich mehr auf schwierige Verkehrssituationen konzentrieren (vgl. [CRZ11, S. 66]). Worauf der Mensch die Aufmerksamkeit legt, wird meist über Reize gesteuert. Eine starke Aufmerksamkeitswirkung bekommen beispielsweise verändernde Reize wie Warnlichter oder Warntöne. Bei diesem Ansatz, auch *bottom-up* genannt, reagiert der Fahrers unwillkürlich und ohne willentliche Anstrengung. Auf der anderen Seite existiert der *top-down* Ansatz, bei dem der Fahrer die Aufmerksamkeit willentlich steuert und sich beispielsweise dem Navigationssystem zukehrt (vgl. [Met09, S. 49ff]). Solchen fahrirrelevanten Objekten wendet sich der Fahrer im Durchschnitt allerdings auch nur 1,6 Sekunden zu, da eine längere Dauer als unangenehm empfunden wird (vgl. [KV11, S. 29f]). Somit wird die These der Europäischen Kommision zur Empfehlung, wie in 2.4.1 beschrieben, nochmals unterstützt.



3 Konzeption

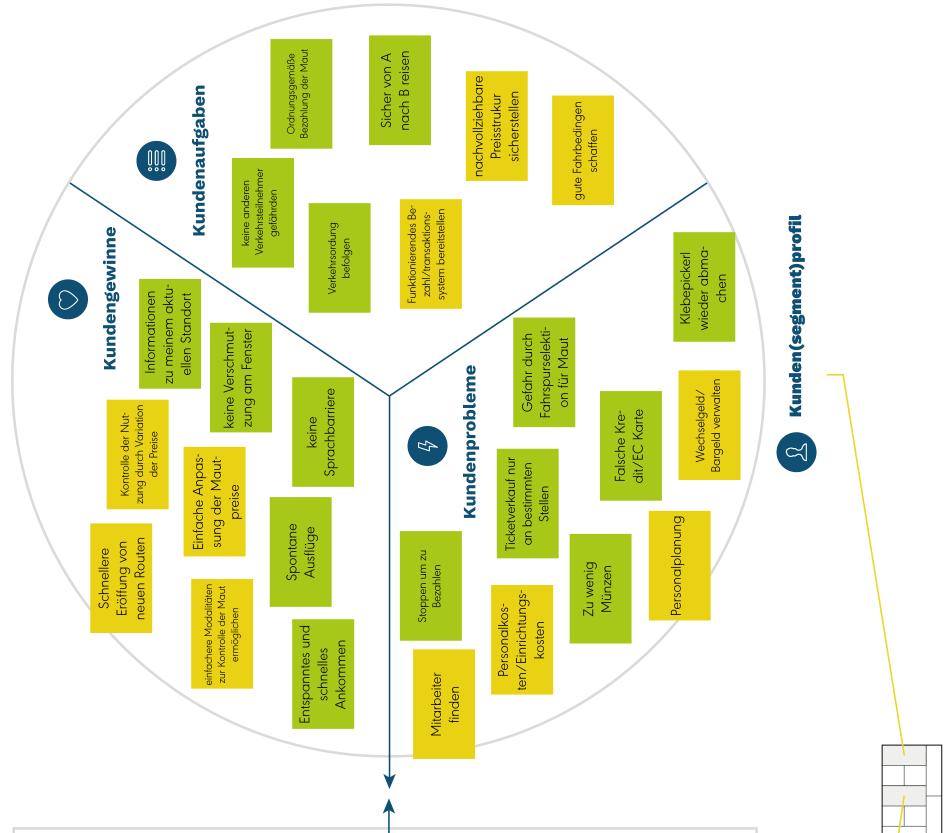
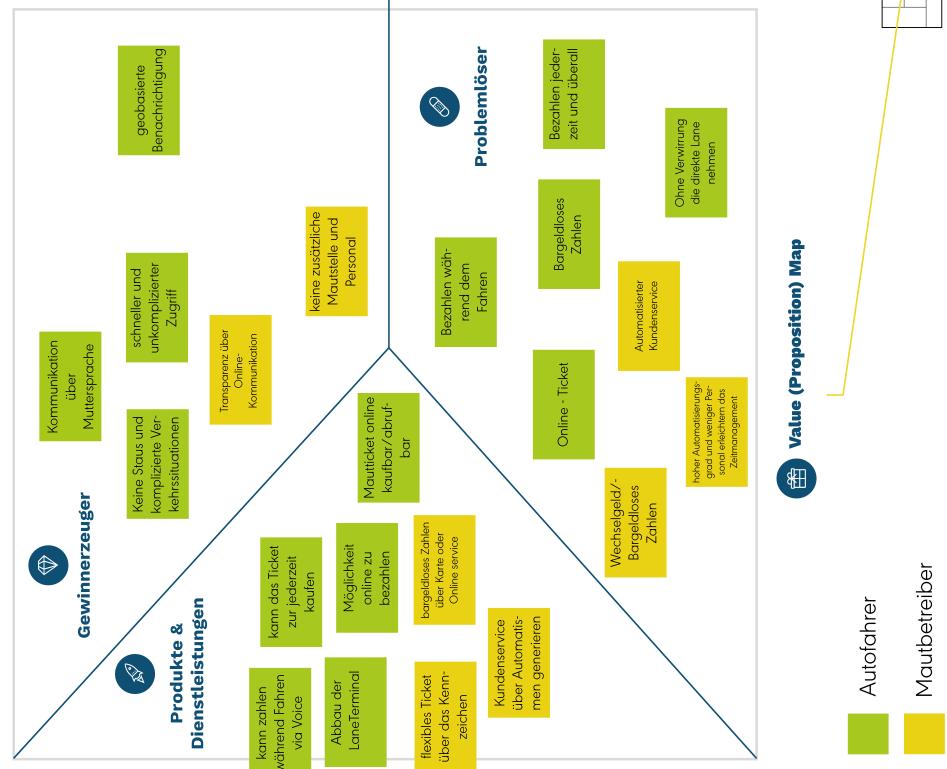
Im folgenden Kapitel wird die Konzeption des Picture Passcode näher erläutert. Es wird zunächst über die Anforderungsermittlung mittels des Value Proposition Canvas VPC diskutiert, um die Perspektive des potentiellen Kunden in der Konzeption aufzugreifen und einzubinden. Anschließend wird das Picture Passcode Verfahren erforscht, um Aspekte in Bezug der Datensicherheit und Ergonomie zu gewährleisten. Zuletzt wird das Conversational Design für den Passcode erörtert.

3.1 Value Proposition Canvas

Das *Value Proposition Canvas* ist die Ergänzung zum *Business Model Canvas*, welche die Komponenten der Kundensegmente und Wertversprechen erläutern sollen. Ziel des VPC ist den Mehrwert für den Kunden zu analysieren [OPBS15]. Es existieren zwei Kategorien, die Value(Proposition)Map und das Kunden(segment)profil, wie in 3.1 zu sehen. Die Value (Proposition) Map dient dazu, das Leistungsversprechen für Pains und Gains der Kunden zu kreieren. Das Kunden(segment)profil wiederum beschreibt ein spezielles Kundensegment. Das Kunden(segment)profil wird in Aufgaben, Probleme und Gewinne für den Kunden aufgeteilt (vgl. [OPBS15, S. 8ff]).

3 Konzeption

Value Proposition Canvas



Balmer Node Canvas

Abbildung 3.1: Value Proposition Canvas (eigene Darstellung in Anlehnung an [OPBS15])



3 Konzeption

Das vorgestellte *Value Proposition Canvas* bezieht sich auf den Kontext für das Zahlen während der Fahrt. Um einen geeigneten Use-Case in Bezug dessen zu betrachten, wurde das Szenario für die Bezahlung eines Mauttickets gewählt. Dabei stellen sich der Autofahrer und Mautbetreiber als Zielgruppe für das Szenario heraus. Autofahrer sind mit der grünen Farbe und Mautbetreiber mit der gelben Farbe im VPC gekennzeichnet. Zunächst wird zwischen den jeweiligen Aufgaben, Problemen und Gewinnen der Zielgruppe differenziert. Es ist zu erwähnen, dass Kundenaufgaben die Voraussetzung bieten um Gewinne zu erzielen und zur Lösung von Kundenproblemen beitragen. Im dargestellten *Value Proposition Canvas* ist der Autofahrer verpflichtet die Verkehrsordnung zu befolgen, keine anderen Verkehrsteilnehmer zu gefährden und sicher von A nach B zu fahren. Außerdem sollte er die Bezahlung der Maut ordnungsgemäß vorrichten. Der Mautbetreiber wiederum, muss für den Mautbenutzer eine nachvollziehbare Preisstruktur und ein funktionierendes Bezahl-/Transaktionssystem sicherstellen. Zudem sollte er gute Fahrbedingungen schaffen, um einen sicheren Fahrt für den Autofahrer zu gewährleisten.

Bei der Bewältigung der Aufgaben entstehen Kundenprobleme, die den Prozess bei der Erfüllung der Aufgabe verlangsamen oder verhindern. Zunächst lässt sich feststellen, dass viele Fahrer aus geschäftlichen Gründen, um in den Urlaub zu fahren oder Verwandte oder Freunde zu besuchen, Ländergrenzen überschreiten. Da die Benutzung der Straßen in vielen Ländern kostenpflichtig ist, müssen Autofahrer ein Mautticket kaufen. Dabei existieren unterschiedliche Mautsysteme, die den Fahrer zwingen, einen Stopp, entweder an einer Raststätte oder an verschiedenen Maut-Terminals auf der Autobahn, einzulegen. [Hän18]. Zusätzlich dazu, können sich die Systeme von Land zu Land unterscheiden. Zu wenig Bargeld, keine Annahme von EC-Karten oder der Verkauf der Tickets ausschließlich an bestimmten Stellen, führen zu vermeidbaren Unterbrechungen der Reise. Darüber hinaus wird die Verkehrssicherheit für den Fahrer negativ beeinflusst. Für ein System bei dem das Mautticket an einem Terminal auf der Autobahn gekauft wird, besteht die Gefahr eines Unfalles. Wenn der Fahrer die Spur wechseln muss, um zum jeweiligen Terminal seines Fahrzeugtypes zu gelangen, kann es zu einer schwierigen Verkehrssituation kommen, da er neben dem Fahren mit hoher Wahrscheinlichkeit nach Geld für die Maut sucht. Auch der Mautbetreiber hat mit der Entwicklung von expliziten Schwierigkeiten zu kämpfen. Er muss zunächst Personal für die jeweiligen Mautstellen organisieren. Dadurch entstehen Personalkosten und zusätzlicher Zeitaufwand, da diese auch zu bestimmten Zeiten eingeteilt werden müssen. Da der Mautbetreiber ein Bezahlssystem sicherstellt, auch für die Barzahlung, muss er



3 Konzeption

die Verwaltung von Wechselgeld und Bargeld gewährleisten. Zuletzt entstehen Kosten für die Einrichtung und Entstehung von Mautstellen für den Mautbetreiber.

Neben Problemen lassen sich auch Aspekte ausweisen, die von der Zielgruppe als gewinnbringend angesehen werden. Dabei wäre für den Fahrer zunächst wichtig, spontane Ausflüge, ein entspanntes und schnelles Ankommen am Zielort und Informationen zum aktuellen Standort, zu ermöglichen. Außerdem sollte die Sprachbarriere, die Kommunikation zwischen Fahrer und Mautbetreiber, nicht erschweren. Für den Mautbetreiber wäre es wiederum wünschenswert, eine einfache Anpassung der Mautpreise und eine schnellere Eröffnung von neuen Routen zu arrangieren. Zudem könnte die Kontrolle der Maut, durch einfachere Modalitäten ermöglicht und die Kontrolle der Straßenbenutzung durch die Anpassung von Preisen geregelt werden.

Im Bereich der Value (Proposition) Map werden Problemlöser zu den jeweiligen Problemen der Zielgruppe vorgestellt. Außerdem werden Gewinnerzeuger, welche Kundengewinne ermöglichen und bestimmte Produkte und Dienstleistungen, die den Kunden bei der Aufgabenerfüllung unterstützen, näher erläutert. Der Fahrer ist gezwungen zu stoppen, um ein Mautticket zu kaufen, wie im oberen Teil beschrieben. Dies könnte durch die Möglichkeit der Bezahlung während der Fahrt verhindert werden. Für den Mautbetreiber kann das Problem der Verwaltung von Bargeld oder Wechselgeld, durch bargeldloses Zahlen gelöst werden. Der Gewinnerzeuger für den Fahrer beispielsweise wäre die Vermeidung von Staus und komplizierten Verkehrssituationen, um ein entspanntes und schnelles Erreichen des Zielortes des Fahrers einzurichten.

Zuletzt kann für Produkte und Dienstleistungen die Möglichkeit gegeben werden, das Mautticket online zu zahlen. Eine weitere Option wäre das Zahlen während der Fahrt via Sprache, damit der Fahrzeugführer sicher und ordnungsgemäß zum Zielort kommt. Zwar ist das Zahlen im Auto, wie in 2.1 beschrieben, bei einigen Autosystemen möglich, allerdings werden alle Bezahlvorgänge ohne weiteren Autorisierungsschritt verrichtet. Somit kann nicht garantiert werden, dass nur eine berechtigte Person die Zahlung auslösen kann. Zusätzlich ist der Schritt der Bezahlung während der Fahrt oft nicht möglich, da die Systeme nicht darauf ausgelegt sind, sondern im Stehen. Um diese resultierenden Probleme zu lösen, bedarf es einen alternativen Prozess, der den Autorisierungsschritt beinhaltet und das Zahlen während dem Fahren ermöglicht.

Das vorgestellte *Value Proposition Canvas* diente zur Unterstützung für die Konzeption des Picture Passcodes über ein multimodales Conversational Interface. Hierbei wurden die essentiellen Voraussetzungen und Bedürfnisse für den Fahrer erläutert.



3 Konzeption

3.2 Konzeptentwicklung des Picture Passcode

Aus dem VPC wurde abgeleitet, dass es ein System bedarf, um das Zahlen während der Fahrt mit dem Aspekt einer Autorisierung, zu ermöglichen. Aufgrund dessen wurden Kriterien aufgestellt, die ein System zu erfüllen hat. Zunächst ist es von Bedeutung, dass das Verfahren eine geringe Angriffsfläche für verschiedene Attacken bietet, um den Zugriff auf die Autorisierungsdaten zu vermeiden. Da sich eine weitere Person im Auto befinden kann, muss explizit die Shoulder-Surfing, Brute-Force und Dictionary-Attacke betrachtet werden. Dabei muss das Risiko minimiert werden, dass ein Angreifer durch reines Beobachten, durch Raten oder der Wahrscheinlichkeit durch Probieren, Zugriff auf die Daten erhalten kann.

Des Weiteren muss das Verfahren ergonomische Faktoren erfüllen, damit es während der Fahrt bedienbar bleibt. In Kapitel 2.4.1 wurde hierzu beschrieben, welche Kriterien dafür nötig sind. Hierbei wurde beispielsweise auf die Interaktion zwischen Fahrer und System geachtet und wie ein Verfahren gestaltet werden muss, damit das Unfallrisiko nicht erhöht wird. Hierbei wurde das Picture Password, in 2.3.1.3 beschrieben und als Inspiration verwendet.

3.2.1 Ausgangspunkt Blackberry Picture Password

Das Picture Password von Blackberry bildet den Ausgangspunkt der vorgestellten Konzeption des Picture Passcodes. Der Grund weshalb die Wahl auf das Picture Password gefallen ist, wird im Folgenden nun erläutert.

Beim Picture Password handelt es sich um ein grafisches Passwort, wie in Kapitel 2.3.1.3 erläutert. Ein grafisches Passwort für die Autorisierung einer Zahlung während der Fahrt zu verwenden, eignet sich als eine effiziente Weise seine Daten zu schützen, da es zunächst einprägsam ist. Eine geringe mentale Beanspruchung ist hierbei die Folge, welche die Fahrweise weniger beeinträchtigen kann. Dies kann es zu einer geringeren Wahrscheinlichkeit von Unfällen führen. Außerdem lässt sich die Eingabe eines grafischen Passwortes während der Fahrt einfacher umsetzen, da bei alphanumerischen Passwörtern eine manuelle Eingabe über das Touch-Display zu einer längeren Blickabwendung von der Fahrsituation führt, wie in Kapitel 2.2 beschrieben. Zwar wäre eine Eingabe über Sprache möglich, jedoch ist die Handhabung für den Benutzer



3 Konzeption

umständlich, wenn er ein acht bis zehn stelliges Passwort mit Sonderzeichen, groß- oder kleingeschriebenen Buchstaben und Umlauten, auszusprechen hat. Außerdem kann die Sicherheit der Autorisierungsdaten nicht mehr gewährleistet werden, wenn sich eine Person in der näheren Umgebung befindet, da bei der Eingabe das Problem des Shoulder-Surfing auftritt. Die Verwendung eines Besitztums für die Autorisierung, erweist sich zudem als unvorteilhaft, da es einfach zu missbrauchen ist. Wenn beispielsweise ein Smartphone nur durch die Anwesenheit als Hardware-Key verwendet wird, kann sich eine weitere Person im Auto Zugriff verschaffen, wenn sich der Besitzer nicht im Auto befindet. In 2.3.2 wurde außerdem beschrieben, dass in der Regel ein zweiter Faktor für Autorisierung benutzt wird. Da dieser ähnlich wie beim TAN-Verfahren über das Smartphone verwendet werden müsste, eignet sich dieses nicht, da die Benutzung des Smartphones während dem Fahren verboten ist.

Zuletzt ist die Überlegung eines biometrischen Verfahrens, welches sicherheitstechnisch und anwenderfreundlich, wie einem Fingerabucksensor, umsetzbar wäre. Allerdings müsste in diesem Fall eine zusätzliche Hardware eingebaut werden, was wiederum Kosten und Zeitaufwand verursachen würde. Da in vielen Fahrzeugen mittlerweile ein Display in der Mittelkonsole verbaut wird, ist die Wahl des Verfahrens schlussendlich auf eine grafische Vorgehensweise gefallen. Der Hauptgrund weshalb das Picture Password als Ansatz favorisiert wurde, waren die sicherheitstechnischen Aspekte in Bezug auf die möglichen Attacken zum Autorisierungsverfahren, wie in 2.3.1.3 nachzulesen.

Unterschiede zum Blackberry Picture Password

Es müssen einige Unterschiede gemacht werden, um den Picture Passcode während der Autofahrt nutzbar zu machen. Das Picture Password wird für das Entsperren des Smartphones verwendet. Der User benutzt also ein Display, welches in naher Distanz über Touch steuerbar ist. Das Bedienen des Displays über manuellen Input wäre während der Fahrt allerdings nicht von Vorteil, da dies, wie in 2.4.1 erläutert, eine zusätzliche Ablenkung von der primären Fahraufgabe darstellt. Aus diesem Grund muss eine Alternative zur Eingabe des Passwortes erfolgen. Des Weiteren muss beachtet werden, dass das Design des Interfaces für die Anwendung reduzierte aber relevante Information beinhaltet, damit eine geringe mentale Beanspruchung und visuelle Ablenkung entsteht. Zuletzt, sollte das Interface des Autorisierungsverfahrens in Bezug zum

3 Konzeption

Layout responsiv sein, damit es an verschiedene Größen und Seitenverhältnisse angepasst und in vielen Autos verwendet werden kann.

3.2.2 Konzeptionsphase

Um den Passcode an den besagten Kontext anzupassen, wurde zunächst, ähnlich wie beim Picture Password, ein Foto als Basis gewählt. Damit keine freien Flächen entstehen können, wurde darauf geachtet, dass eine ausreichende Anzahl an markanten Positionen zur Verfügung gestellt wurden. Dabei haben sich Fotos von zentralen Plätzen bzw. Städten der Welt, wie dem Times Square in New York oder die Shibuya-Kreuzung in Tokyo, als eine vielversprechende Option herausgestellt.



Abbildung 3.2: Auswahl der Position für den Picture Passcode (Quelle-Hintergrund: [New09])

Da der Autorisierungsprozess im Ursprung über den Touchscreen eines Blackberry-Smartphones ausgeführt wird, muss dieser anders definiert werden. Ein Touch-Display reagiert sensiv auf Berührungen, sodass kleinste Verschiebungen ohne Anstrengung vollbracht werden können. So kann ein Raster bzw. eine Zahl an sämtliche Position verschoben werden. Dies ist bei einem VUI nicht möglich, da keine komplexen Verschiebungen über Sprachkommandos praktikabel sind. Als Problemlösung wurde die Überlegung angestellt, nur die Zahl auszusprechen, welche im Raster auf der gewählten Position erscheint. Dadurch müsste sich der User nur die Position einprägen und wäre nicht mehr an die Zahl gebunden. Um alle Positionen im Foto abzudecken, wäre allerdings ein Raster mit vielen Zahlen nötig. Ein solches Raster eignet sich in diesem Fall jedoch nicht, da eine größere Distanz zum Display gegeben ist. Somit würde ein einfacher schneller Blick nicht ausreichen, um die Übersicht zu erhalten.

3 Konzeption



Abbildung 3.3: Markante Positionen mit Zahl (Quelle-Hintergrund: [New09])

Deshalb gab es den weiteren Ansatz, Zahlen auf unterschiedliche markante Stellen im Foto zu legen. Demnach wird die Entstehung von leeren Stellen vermieden, da jede Position als ein *Hot-Spot* angesehen wird und diese gleichwertig zu jeder anderen Stelle behandelt werden kann. Im Registrierungsprozess hat der User nun die Möglichkeit, eine Position auf dem Foto auszuwählen. Die Position hat eine *Sensitive-Area*, sodass der User im Autorisierungsprozess weiß, dass der Punkt einen Toleranzbereich besitzt. Somit kann die Zahl eine geringe Positionsabweichung von der ursprünglichen Stelle aufweisen, wie in 3.2 zu sehen. Bei diesem Verfahren wird das Problem des Shoulder-Surfings wieder deutlich, da ausgeschlossen werden kann welche Positionen nicht als *Secret*, also gewählte Position, möglich sein kann. Da für den Picture Passcode nur ein Faktor zur Sicherheit gegeben war, wurde auf den zusätzlichen Aspekt der Rasterverschiebung zurückgegriffen.

Damit eine Zahl, zu einer bestimmten Position mit wenigen Sprachkommandos verschoben werden kann, musste vorweg die Frage geklärt werden wie viele Zahlen sich in einem Raster befinden können. Es wurde vorweg ein Raster mit einer Anzahl an 16 Zahlen gewählt, die in einem 4 x 4 Raster platziert sind. Die Folge, die sich jedoch daraus ergibt, ist die limitierte Wahl von Stellen im Foto, die sich der User aussuchen kann. Da der Großteil, wie in Kapitel 2.3.1.3 erläutert, dazu neigt eine prägnante Stelle zu wählen, kann es wieder zu einer Entstehung von Hot-Spots führen, da einige Stellen markanter sind als andere.

Nachdem sich der User für eine Position und Zahl entschieden hat, folgt die Autorisierung. Hierbei war die Überlegung, die Verschiebung des Rasters, wie schon im oberen Teil angedeutet, mithilfe von Kommandos zu regeln.

3 Konzeption



Abbildung 3.4: Platzierung des Zahlenrasters (Quelle-Hintergrund: [New09])

Der User hat in diesen Fall, die Kommandos *links*, *rechts*, *unten* oder *oben* zur Verfügung. Wenn die Zahl 4, siehe 3.4, nun auf die Zielposition, siehe 3.2, verschoben werden soll, verwendet der Benutzer das Kommandorechts. Eine Abfolge von Kommandos, wie *links*, *nach unten* wäre auch eine Option. Allerdings müsste ein weiterer Hinweis für die Beendigung der Sequenz gegeben werden. Als zusätzlicher Sicherheitsfaktor wird das Raster nicht dynamisch angepasst, damit es ein externer Beobachter schwerer hat die Verschiebung des Rasters zu verfolgen. Dieses Verfahren birgt jedoch drei Schwierigkeiten mit sich. Zunächst kann sich eine andere Person durch Probieren zufällig Zugriff auf die Daten verschaffen, ohne Position oder die Zahl kennen zu müssen. Wenn der Angreifer alleinig das Kommando *rechts* verwendet, kann die Zahl nur durch Zufall auf die Zielposition gelangen, wenn der Befehl nicht aktiv durch eine Formel abgeschlossen werden muss. Des Weiteren kann über das Ausschlussverfahren erkannt werden, welche Zahl, zu welcher Position gehört. Dabei geht der Angreifer Position für Position durch und beobachtet, welche Zahl mehr als einmal auf einer Stelle gelandet ist. Wenn beim Autorisierungsprozess unterschiedliche Zahlen an eine Stelle geschoben werden, kann diese als nicht relevant angesehen und ausgeschlossen werden. Zuletzt ist die Anpassungsfähigkeit des Fotos an verschiedene Displaygrößen als gering, und damit als Problemfall einzustufen. Dadurch kann je nach Displays nur ein bestimmter Ausschnitt angezeigt werden, sodass die Auswahl der Position effektiv eingeschränkt wird. Außerdem können nur wenige Sprachkommandos verwendet werden, wenn nur eine kleine Matrix an das Display angezeigt werden kann.

3 Konzeption

Somit hat sich herausgestellt, dass ein Foto nicht als geeigneter Hintergrund dient. Folglich bot sich ein Hintergrundbild an, welches an ein Raster oder eine Matrix erinnert. Es wurde zunächst an eine Collage von Briefmarken oder Bildern von Wahrzeichen gedacht, sodass eine Art von Setzkastensystem entstehen konnte.



Abbildung 3.5: Collagen mit Briefmarken und Wahrzeichen (Quelle:[Del15] [Bri])

Dadurch konnte das Zahlenraster, wie im oberen Teil beschrieben, bestehen bleiben. Zudem wurde die Problematik der Hotspots eliminiert, da jedes Bild gleichwertig behandelt werden kann. Alternativ zur einer Briefmarkencollage kann das Setzkastensystem mittels erstellter Vektorgrafiken bzw. Icons herbeigeführt werden. Dies birgt den Vorteil der Flexibilität um eine dynamische Anzahl von Icons zu produzieren und zu verwenden. Diese können zusätzlich individuell gestaltet werden, um eine Reduktion von Details zu erzielen. Zwar wurde ein geeigneter Hintergrund als Basis für das Verfahren ausfindig gemacht. Allerdings bestand das Problem der Rasterverschiebung weiterhin, weshalb ein neues Vorgehen für den Picture Passcode erörtert wurde.

3.2.3 Funktionsweise des Picture Passcode

Grundlegend besteht der Picture Passcode aus einem dreistelligen Zahlencode. Um diesen zu erhalten wurde zunächst ein Set mit acht Icons generiert. Diese werden in einer 4×2 Matrix dargestellt. In einem Beispiel besteht das Set aus Icons mit Wahrzeichen, siehe 3.6. Der User muss sich nun ein Icon des Sets aussuchen. Das gewählte Icon des Benutzers wird im System als dessen *Secret* registriert, da es essentiell für das Erlangen des Passcodes ist. Beim Autorisierungsverfahren weist das System nun jedem Icon, zwei unterschiedliche Zahlen zu, die zu einem Passcode zusammengesetzt werden. In diesem Fall hat auch das *Secret*, welches vorher ausgewählt wurde, zwei Zahlen erhalten. Somit entstehen zwei Raster, die jeweils unterschiedliche Zahlen

3 Konzeption

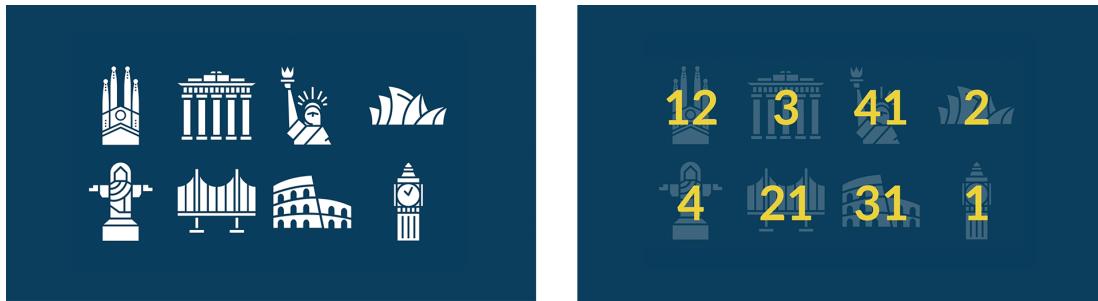


Abbildung 3.6: Icon Set und Wechsel der Zahl im Sekundentakt

beinhalten. Die Besonderheit, die nun folgt, ist die unterschiedliche Farbe der Zahlen, der jeweiligen Raster. Dabei haben die Zahlen eines Rasters, die Farbe gelb und die Zahlen des anderen Rasters, die Farbe weiß, erhalten. In diesem Zusammenhang, gilt die gelbe Zahl als ersten Teil und die weiße Zahl als zweiten Teil des Passcodes. Wenn nun beispielsweise die "12" als erste Zahl und die "3" als zweite Zahl zusammen gesetzt werden, bildet sich der Passcode "123". Der User muss sich bei diesem Verfahren nicht die Zahl bzw. den Passcode merken, sondern nur das *Secret*. Somit ist der Passcode nur ein Adapter auf das *Secret*, weshalb in erster Linie nicht darauf geschlossen werden kann.



Abbildung 3.7: Zusammengefügter Passcode

Weiterhin ist beim Autorisierungsprozess zu beachten, dass die Zahlenraster sekündlich ihre Anzeige wechseln. Mit anderen Worten, erscheint zunächst das gelbe Zahlenraster, das über die Icons gelegt wurde. In der nächsten Sekunde wechselt die Anzeige zum weißen Zahlenraster. Als Gedankenstütze wird noch ein zusätzlicher Layer angezeigt, der nur die Icons ohne Zahlen aufführt. Somit hat der User nochmal die Möglichkeit, die Position des *Secret* ausfindig zu machen. Zusammenfassend werden die drei Layer im Sekundentakt umgeschaltet, sodass eine dreisekündige Sequenz entsteht, die in einer Schleife ohne Ablauf einer Zeit durchgespielt wird, siehe 3.6.

3 Konzeption

Die Sicherheit in Bezug des Picture Passcodes, wird durch die Wahl des Zahlenspektrums und die Kombinationsmöglichkeiten der Zahlen definiert. Da der Passcode aus einem dreistelligen Zahrencode besteht, wurden für beide Layer insgesamt acht zweistellige Zahlen und acht einstellige Zahlen definiert, die aus einer Range von vier Ziffern zusammengestellt werden. Im besagten Beispiel wurden Zahlen aus der Ziffermenge [1, 2, 3, 4] gebildet. Diese können sich für jeden Autorisierungsprozess unterscheiden, sodass auch andere Ziffern verwenden können, um ein größeres Zahlspektrum zu generieren. Da sich der Passcode aus einer zweistelligen und einstelligen Zahl zusammensetzt, können zwei Kombinationsmöglichkeiten zum Passcode führen. Lautet der Passcode beispielsweise "123" kann dieser zunächst aus den Zahlen "1" und "23" oder "12" und "3" kombiniert werden. Mithilfe der unterschiedlichen Farben wird noch ein zusätzlicher Sicherheitsfaktor eingebettet, sodass nicht direkt auf das *Secret* geschlossen werden kann. So können die Zahlen "1" und "23" auf einem anderen Icon auftauchen, jedoch wird hierbei der falsche Passcode "231" gebildet, da die Reihenfolge der Farbe beachtet wird. Dementsprechend wurden also zwei Sicherheitsfaktoren hinzugefügt, die versuchen das *Secret* zu schützen.

Es lässt sich jedoch feststellen, dass die Wahrscheinlichkeit, das *Secret* zu identifizieren bei $> 1/8$ liegt. Das liegt daran, dass der Angreifer nach einiger Zeit auf den richtigen Passcode stößt, wenn dieser jedes Icon systematisch durchgeht und verschiedene Kombinationsmöglichkeiten testet. Um dieses Problem zu lösen, wurde ein zusätzliches Icon-Set hinzugefügt. Der User wählt in diesem Verfahren nun zwei, statt einem *Secret*. In diesem Zusammenhang, wird ein weiterer Layer zu der Sequenz, hinzugefügt. Der Layer ist wiederum eine zusätzliche Gedankenstütze des zweiten Icon-Sets ohne Zahlen. Somit entstehen vier Layer, die sekündlich auf dem Interface wechseln.



Abbildung 3.8: Picture Passcode Icon-Sets und Auswahl der Secrets

3 Konzeption

Der zusätzliche Icon-Layer setzt die Wahrscheinlichkeit von $> 1/8$ des Identifizierens der *Secrets* auf $> 1/64$ herunter. Dies ergibt sich daraus, dass $n = 4$ (Anzahl der Ziffernmenge) und $r = 2$ (Anzahl der Zahlkombinationen), die durch Permutation $(n^r) * n * 2$ definiert werden. Es ist zu beachten, dass die n^r zu Beginn für die Zweizifferkombination und das $n * 2$ für die Kombination zum dreistelligen Passcode steht, die auf zwei verschiedene Wege entstehen kann. Mit diesem Verfahren lassen sich also 128 Kombinationen generieren, sodass endgültig 64 wählbare Passcodes existieren.

Durch den zusätzlichen Layer muss ein Angreifer auf zwei *Secrets* schließen, um sich Zugriff auf die Daten zu verschaffen. Unabhängig vom Passcode werden noch weitere Sicherheitsfaktoren eingebaut. Ähnlich wie beim PIN, siehe 2.3.1.1 hat der Benutzer drei Versuche, um auf den richtigen Passcode zu schließen. Falls alle Versuche inkorrekt waren, sperrt das System den Zugriff auf die Daten. Nachdem der User den Passcode gedanklich zusammengefügt hat, kann er dem System zunächst mitteilen, dass er den Passcode erkannt hat. Darauffolgend wechselt das System zu einer anderen Anzeige. So wird ein weiterer Faktor für Sicherheit der *Secrets* hinzugefügt, da der Angreifer nicht mehr nachzusehen kann, welche Zahlen auf welchen Icons erschienen sind. Danach kann der User dem System, den Passcode als Aufzählung oder als ganze Zahl mitteilen.



Abbildung 3.9: Anzeige zur Passcode Eingabe und erfolgreiche Autorisierung



3 Konzeption

3.3 Ergonomie

Zwar beinhaltet der Picture Passcode ergonomische Aspekte, welche die Verwendbarkeit dessen während der Fahrt ermöglicht. Allerdings muss explizit auf bestimmte Vorgehensweisen eingegangen werden, die essentiell sind, um die Sicherheit im Straßenverkehr zu gewährleisten. Dabei müssen Bedingungen geschaffen werden, die sowohl die Sicherheit des Fahrers, als auch der anderen Straßenverkehrsteilnehmer, garantieren. Aus diesem Grund wird zunächst auf die Gestaltung des Interfaces eingegangen. Es wird die Vorgehensweise der Kommunikation zwischen Fahrer und System diskutiert. Daran anknüpfend, wird auf die gestalterischen Aspekte des Picture Passcode eingegangen, um eine bessere Usability für den Fahrer sicher stellen zu können.

3.3.1 Interfaces

Bei der Gestaltung des Interfaces wurde sich an den Empfehlungen bzw. Punkten, die in 2.4 beschrieben sind, orientiert, um ein fahrtaugliches System zu implementieren. Wie in Kapitel 2.2 beschrieben, gilt hierbei die Devise, dass die Eingabe des Picture Passcode über Voice erfolgen soll. Unterdessen soll der Sprachassistent des Autos verwendet werden, sodass eine Vereinheitlichung der Bezahlung bei verschiedenen Anbietern mit einem Prinzip möglich ist. In Bezug auf das GUI, gibt es unterschiedliche Bereich des Cockpits, welche sich als geeignetes Interface innerhalb eines Fahrzeuges anbieten würden. Die Auswahl lag zwischen einem Display in der Mittelkonsole FIS, einem Head-Up oder Dashboard-Display FAS. Hierbei konnte das Head-Up-Display als mögliches Interface ausgeschlossen werden, da dieses meist zu wenig Kontrast bietet, um eine geeignete Darstellung des Picture Passcode Verfahrens zu garantieren. Im Gegensatz dazu, bietet das Display in der Mittelkonsole, genügend Kontrast und zusätzlich den Vorteil der Verbreitung, da dieses bereits bei vielen Autos verbaut ist. Ein Dashboard-Display bietet wiederum den Vorteil, dass andere Personen nur eine eingeschränkte Sicht auf das Interface erhalten. Außerdem wird die Blickabwendung zur Fahrbahn gering gehalten, da ein kurzer Weg vom Dashboard-Display zur Verkehrssituation, benötigt wird.



3 Konzeption

3.3.2 Gestalterische Aspekte

An erster Stelle wurde auf die Gestaltung der Icons eingegangen. Diese wurden so abstrakt wie möglich gehalten, jedoch so, dass sie einfach zu unterscheiden sind. Das soll dazu führen, dass sie für den Fahrer während der Autorisierungsphase schnell zu erkennen sind. Für die jeweiligen Icon-Sets wurde ein bestimmtes Thema festgelegt, wie Wahrzeichen oder Landschaften. Der User kann somit schneller differenzieren, welches Secret er in Bezug des dargestellten Icon-Sets, suchen muss. Die Icons können dynamisch angepasst werden, ohne einen Qualitätsverlust zu erleiden. Das birgt den Vorteil, der Anpassungsfähigkeit für unterschiedliche Displaygrößen. Im Ursprung wurden die Icons in einer 4 x 2 Matrix, wie im oberen Teil beschrieben, angepasst. Allerdings lassen sich diese abhängig von der Displaygröße und Format, auch in unterschiedliche Rastergrößen implementieren. Die Zahlen, die beim Autorisierungsprozess über den Icons liegen, wurden kontrastreich und mit einer prägnanten Farbe versehen. So können diese leichter zu erkennen und differenziert werden. Dafür wurde über die Icon-Sets ein blau-halb-transparenter Layer gelegt, um den Kontrast der Icons herauszunehmen. Die Farbe der Zahlen wurde so gewählt, dass sie sich vom Hintergrund abheben und sich zugleich unterscheiden. Es wurde auch die Überlegung angestellt, prägnante Farben wie rot oder grün, unabhängig vom Raster, für die Zahlen zu wählen. Dadurch geraten bestimmte Zahlen in den Vordergrund, da sie sich zu sehr von anderen Zahlen mit einer neutraleren Farbe abheben. Deshalb wurde die Farbe weiß und gelb für jeweils ein Zahlenraster gewählt, da sich diese genug unterscheiden und sich nicht zu sehr in den Fokus drängen. Das grafische Aussehen der Zahlen beeinflusst auch die Erkennbarkeit der jeweiligen Nummern. Für bestimmte Formen unterschiedlicher Zahlen, können sich einige schwerer unterscheiden, wie in der Abbildung 3.10 zu sehen ist.

Wie eine Zahl verwechselt werden kann, hängt von den Unterscheidungsmerkmalen der jeweiligen Zahlen ab. Je weniger Merkmale vorhanden sind, desto schwieriger ist es diese voneinander zu unterscheiden (vgl. [Mis14]). Außerdem ist zu beachten, dass es wahrscheinlicher ist eine Zahl zu erkennen, die weniger Merkmale aufweist, als die ähnliche Gegenzahl. Die Nummer 8 und 3 sind in ihrer Form zwar ähnlich, jedoch enthält die 3 weniger Unterscheidungsmerkmale, als die 8. Deshalb wird die Zahl weniger wahrgenommen [KB81] (vgl. [Mis14]). Die Vermeidung einer Verwechslung von Zahlen hängt von der Leserlichkeit, Lesbarkeit und Sichtbarkeit der Zahl ab. Bei einigen Zahlen kann die Schreibweise und Schriftart, einen entscheiden Faktor darstellen.

Numeral–Numeral
0 and 8
3 and 9
3 and 8
4 and 9
5 and 8
5 and 3
6 and 8
7 and 1

Abbildung 3.10: Ähnlichkeiten unterschiedlicher Zahlen (Quelle: [Mis14])

Beispielsweise ähnelt sich die 7 mit der 1. Hier wäre die Möglichkeit einen horizontalen Strich in die Mitte der 7 zu platzieren, um diese Zahl unterscheidbarer zu machen (vgl. [Mis14]). Damit der Fahrer die Zahlen trotz dessen einfach unterscheiden kann, fiel die Wahl auf eine seriflose Schrift, da diese einfacher auf elektronischen Interfaces zu lesen ist [Obe13]. Außerdem wurde darauf geachtet, keine digitale Schriftart zu wählen, um die Verwechslungsgefahr weiter einzudämmen. Der Faktor, wie merkbar eine Zahl ist, gilt auch als ein Aspekt, die bei der Zusammenstellung des Passcodes auftritt. Die Vielzahl merkt sich Nummern eher, wenn eine Assoziation zu ihnen hergestellt werden kann (vgl. [Bor17]). Dabei sind Repdigits wie 111 oder 222, aufsteigende und absteigende Zahlen wie 123 oder 987 und Zahlen die mindestens eine 0 enthalten, relevant für den besagten Use-Case. Es soll die Wahl solcher Zahlkombinationen für den Passcode vermieden werden, damit externe Angreifer nicht leicht auf die Zahlen schließen können, die für die Kombination benötigt wird.

3.4 Datensicherheit

Für die Sicherheit der Autorisierungsdaten, wurde für die Übertragung von Daten ein Konzept entwickelt, um automatisierte Angriffe zu vermeiden. Die Funktionsweise ähnelt der Übertragung von biometrischen Daten bei iOS Geräten [Off18], die zur Authentifizierung der eigenen Person dienen sollen. Im Kontext, ist das lokale Systems des Fahrzeuges, ein essentieller Faktor. Das Auto generiert zunächst den Passcode, der zur Autorisierung des Users dienen soll. Wenn der User den Passcode eingegeben hat, prüft das System die Kenntnis der Secrets in Bezug des Passcodes.

3 Konzeption

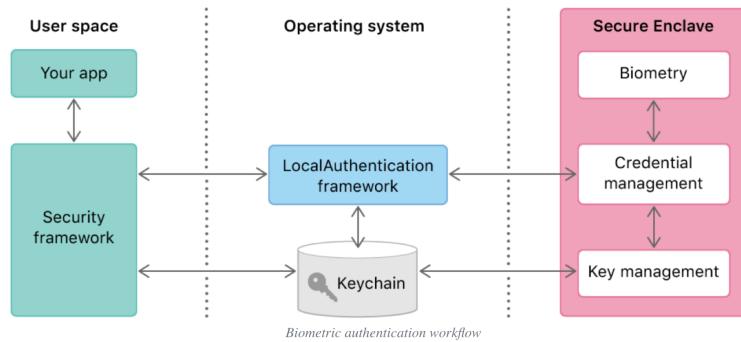


Abbildung 3.11: Biometrisches Authentifizierungsverfahren iOS (Quelle: [Off18])

Bei einer positiven Prüfung erlauben die Secrets, Zugriff auf den Secure-Store des Autos, der einen Private-Key /Zertifikat enthält. Der Private-Key enthält eine Signierung die über einen Public-Key über das HTTPS-Protokoll an den externen Server gesendet wird. Der Server erhält die positive Bestätigung und erlaubt den Zugriff zur Autorisierung einer Zahlung. Der Vorteil eines solchen Systems, ist die Schwierigkeit einen automatisierten Angriffs zu starten, wie beispielsweise einer Brute-Force-Attacke. Der private Schlüssel wird nur im Secure-Store des Wagens gespeichert und nie auf dem externen Server, oder als Datensatz im Arbeitsspeicher gehalten.



4 Prototyping

Das folgende Kapitel behandelt die prototypische Umsetzung einer Fahrsimulation und Conversational Design, um das Szenario *Bezahlen während dem Fahren via dem Picture Passcode* zu imitieren. Zu Beginn wurde der mentale, physische und temporale Task-Load des Fahrens mittels der genannten Fahrsimulation beschrieben. Die sekundäre Aufgabe, dem Zahlen via Picture Passcode, wurde wiederum mithilfe des Conversational Design umgesetzt. Beide Komponenten bilden die Basis, um einen Usability-Test zu initiieren und die Ergebnisse zu evaluieren. Zuletzt wurden funktionale Komponenten implementiert, um einen dynamischen Passcode zu generieren.

4.1 Erstellung eines Prototypen

Prototyping Zyklus

Zunächst wird unter Prototyping, die frühe Externalisierung eines Produktes oder einer Benutzerschnittstelle verstanden, welche durch eine iterative Veränderung evaluiert und verbessert wird. Dies ermöglicht ein schnelles Feedback, um das Risiko für eine späte Erkennung einer Fehlentwicklung des Produktes zu senken ([RF16, S. 72ff]). Dabei wird zwischen drei verschiedenen Stufen eines Prototypen, die einen gewissen Reifegrad erreichen, unterschieden. Der Low-Fidelity-Prototyp (LoFi) ist die einfachste und kostengünstigste Methode, um ein Produkt zu testen. Es wird oft ein Papierprototyp verwendet, welcher mit Papier und Stift skizziert wird, um grafische Oberflächen und deren Funktion zu testen. Ein Mid-Fidelity-Prototyp (MidFi) kommt dem Endprodukt wiederum näher, da bereits interaktive Designs und Oberflächen verwendet werden können. Solche Prototypen erhalten zunächst Mock-Daten, da komplexe und dynamische Funktionen nur vorgetäuscht werden. Zuletzt bietet ein High-Fidelity-Prototyp (HiFi) ausführbare Funktionen an, die der endgültigen Software bzw. Produkt sehr ähnlich sind und ein natürliches Verhalten beim Testen generieren können.



4 Prototyping

(vgl. [Esp18]). Der entwickelte Prototyp dieser Arbeit kann als ein Mid-Fidelity-Prototyp bezeichnet werden.

Prototyping Tool

Der genannte Prototyp wurde mittels der Software Max von der Firma Cycling '74 umgesetzt [Cyc]. Max ist eine grafische Entwicklungsumgebung, bei der es möglich ist unterschiedliche Bearbeitungseinheiten über virtuelle Leitungen zu verbinden. Projekte die in Max implementiert werden, sind in sogenannten *Patchern* organisiert. Diese können sich gegenseitig durch In- und Outputs beeinflussen, sodass mehrere *Patcher* miteinander kommunizieren können. In Max ist es möglich, multimodale Prototypen zu erstellen, weshalb dies als geeignetes Tool für den besagten Prototypen gewählt wurde. Durch vorhandenes Wissen und Komponenten vorheriger Arbeiten, konnte ein schneller Prozess der Entwicklung erfolgen.

4.1.1 Fahrsimulation

Zunächst war es notwendig eine Simulation zu generieren, welche die Beanspruchung auf mentaler, physischer und temporaler Ebene der Fahrzeugführung so gut wie möglich nachstellen sollte, um die Funktionalität des Picture Passcode investigativ zu evaluieren. Hierbei müssen bestimmte Parameter individuell konfigurierbar sein, um flexibel Veränderungen für Test-Cases machen zu können. Der Fahrer soll die Möglichkeit haben nach links, rechts und geradeaus zu fahren. Die Variabilität der Geschwindigkeit des Fahrzeuges sollte jedoch über den Moderator gesteuert werden, um unterscheiden zu können, wie der Fahrer in Bezug auf unterschiedliche Fahrbedingungen reagiert und agiert. Um Anomalien der Fahrweise des Fahrers feststellen zu können, wurde die Bedingung gestellt, dass bei jeder Berührung eines anderen Autos oder der Leitplanke, ein kurzes Signal ertönt, siehe 4.1. Somit kann, für den Fahrer und Moderator, auf eine Kollision hingewiesen werden. Die Anzahl der Kollisionen wird parallel in einen Counter übertragen, da die Fehlerquote für die spätere Evaluation essentiell ist. Als zusätzliche Komponente, wurde die Betätigung des Blinkers in den Prototypen übertragen, um abschätzen zu können, ob der Fahrer Verkehrsregeln durch die Benutzung des Systems beachten kann. Dabei wurden Fehlersysteme entwickelt, welche die Quo-
te messen, ob der Fahrer korrekt, entgegengesetzt gefahren ist oder nicht blinke. Da die Zeit nur für eine prototypische Umsetzung einer Fahrsimulation reichte, stand die

4 Prototyping

Gestaltung der Oberfläche nicht an oberster Stelle. Jedoch sollte sie für den Fahrer nachvollziehbar und klar erkennbar an eine reale Fahrsituation erinnern. Hierbei wurde sich an die erste Version des Spiels *Speed Race* von Tomohiro Nishikado orientiert, welches im Jahr 1974 erschien [Koh05]. Es ist eines der ersten Autorennspiele und fungiert nur über grafische Elemente, die sich von oben nach unten bewegen. Somit wird die Geschwindigkeit des eigenen Fahrzeugs vorgetäuscht, da diese Objekte sozusagen *überholt* werden.

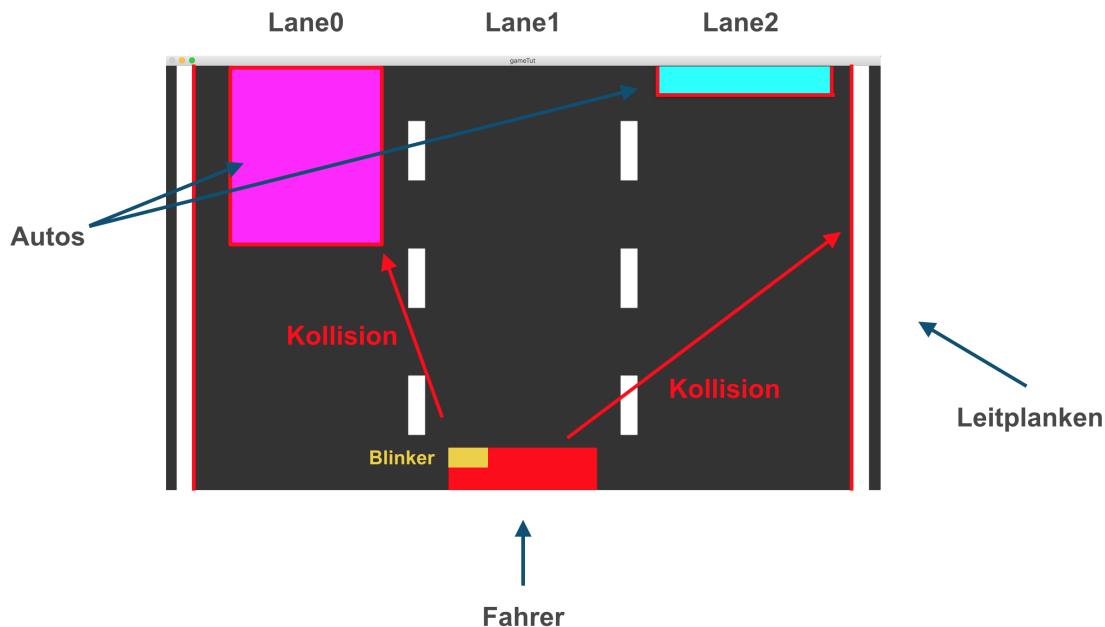


Abbildung 4.1: Grafische Oberfläche der Fahrsimulation in Anlehnung an (Quelle:[Koh05])

Im besagten Prototypen, wie in 4.1 zu sehen, erscheint das Fahrzeug, welches der Fahrer steuert, als rotes Rechteck, angeschnitten, am unterem Bildschirmrand. Dieses kann er gegebenenfalls nach links oder rechts steuern, um den vor ihm fahrenden Fahrzeugen auszuweichen. Das Auto bewegt jedoch nicht nach oben, da es unten fest verankert ist. Der Grund liegt hierfür, wie im oberen Teil beschrieben, bei der Einstellung der Geschwindigkeit. Dabei soll die Geschwindigkeit der vorausfahrenden Fahrzeuge gesteuert werden und nicht des Fahrers. Die Fahrzeuge sind mit den Farben Magenta und Cyan vom Fahrer zu unterscheiden. Des Weiteren wurden Leitplanken bzw. Fahrbahnbegrenzungen eingebaut, welche rechts und links des Bildschirmrandes in weiß zu sehen sind. Zuletzt wurden zwei Fahrstreifen hinzugefügt, um das Gefühl einer dreispurigen Fahrbahn nachzuahmen.



4 Prototyping

Umsetzung der Anforderungen

Für die Erstellung der Fahrsimulation und dessen zusätzliche Komponenten wurde in Zusammenarbeit mit Steffen Blümm, Technical Lead CUI der Firma adorsys, gearbeitet. Um die besagten Anforderungen an den Prototypen umzusetzen, wurde hierbei die *Space Invaders* Anwendung, die komplett in Max umgesetzt wurde [Fod18], als Fundament für den Prototypen verwendet und abgewandelt.

Für die Darstellung der Fahrsimulation, wurde Jitter eingesetzt, welches in Max integriert ist und für die Verarbeitung mehrdimensionaler Matrizen konzipiert wurde, und somit gut für die Verwendung von Video und von 3D-Grafiken in Echtzeit geeignet ist. Für Jitter existiert das jit.world-Objekt, welches die Funktionalität mehrerer Jitter-Objekte kapselt [Jit]. Bezogen auf den Kontext, werden alle grafischen Elemente, die benötigt werden, zusammengeführt und in einem Panel angezeigt. Dieses wird als Koordinatensystem angesehen, dessen Ursprungspunkt 0 für x und y in der Mitte liegt.

Einen Überblick, der Fahrsimulation Logik, kann über die 4.2 entnommen werden. Hierbei ist zu betrachten, dass für die Gestaltung der Fahrbahn, zu Beginn zwei Leitplanken mittels des *gridshape*-Objektes, also eine einfache geometrische Form in Bezug zur Matrix, in den *CarSimulationOSC*-Patcher, eingefügt wurden. Da sie keine weitere Funktionalität benötigen, wurde ihnen nur die Farbe, Skalierung der Größe und Position mitgegeben. Ein ähnliches Prinzip verfolgen die Mittelstreifen, welche die Abgrenzung der Fahrbahnen darstellen. Max bietet die Möglichkeit *Abstractions* für die Ausgliederung von Funktionalitäten, zu erstellen. Diese sind vergleichbar mit Klassen in der objektorientierten Programmierung. Folglich wurde eine eigene *LineEntity-Abstraction* gebildet, welche die Funktionen eines einzelnen Streifens des Mittelstreifens kapselt. Somit wurden vier *LineEntity*-Objekte implementiert, die in einer konstanten Geschwindigkeit und Zeitabständen getriggert werden, um ihre Position von oberhalb bis unterhalb des Fenster zu animieren (*LineMove*). Diese werden in einer Schleife, wenn sie aus dem Displayfenster verschwinden, wieder an ihren Ursprung positioniert, sodass eine konstante Flussbewegung entstehen kann. Dadurch kann dem Fahrer eine konstante Geschwindigkeit vorgetäuscht werden, um eine reale Fahrsituation auf einer Autobahn oder dreispurigen Fahrbahn zu generieren, wie in 4.1 zu sehen.

4 Prototyping

CarSimulationOSC

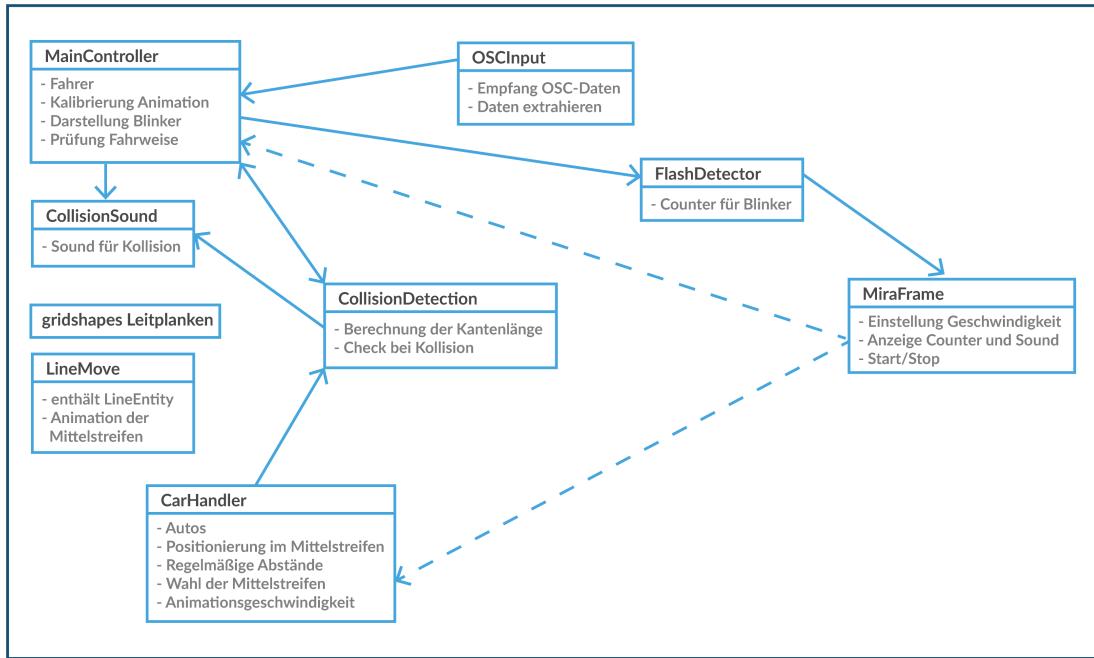


Abbildung 4.2: Hauptpatcher der Fahrsimulation in Anlehnung an (Quelle:[Fod18])

Die Autos, die sich vor dem Fahrer befinden, verfolgen in der Simulation, ähnliche Ansätze der *LineEntity*. Dennoch müssen einige Unterschiede vollzogen werden, da mehr Funktionalitäten verlangt sind. Zunächst werden alle Funktionen der vorausfahrenden Autos erneut in einer *Abstraction* namens *CarHandler* ausgelagert. In diesem *CarHandler* befinden sich wiederum zwei *CarEntities* und ein *SpaceController*. Eine *CarEntity* enthält ein Auto, welches als eine *gridshape* dargestellt wird. Damit beide Autos unterscheidbar sind, wird jeder *CarEntity* eine ID zugewiesen. Bevor eine *CarEntity* getriggert wird, erfolgt eine vorläufige Überprüfung, auf welcher Fahrspur das Auto platziert werden kann. Dabei wird zunächst die *CarPositioning-Abstraction* angesprochen. Diese enthält ein Skript, welches für die Wahl der Fahrbahn verantwortlich ist. Vorweg wurde jeder Fahrbahn eine ID zugewiesen, um diese voneinander unterscheiden zu können, siehe 4.1. Von außerhalb, wird im weiteren Verlauf eine zufällige Fahrbahn vorgeschlagen. Das Skript prüft in einem Dictionary, ob die Fahrbahn belegt ist. In diesem sind alle States der Lanes aufgezeichnet. Da zu Beginn keine Fahrbahn belegt ist, wurden allen Lanes, den State '0', zugewiesen. Dieser gibt an, dass die jeweilige Fahrbahn nicht belegt ist und somit freigegeben werden kann. Wenn sich für eine Fahrbahn entschieden wurde, wird der State, in die ID des jeweiligen Fahrzeugs gewandelt, sodass darauf kein weiteres Fahrzeug positioniert wird.



4 Prototyping

Somit kann zusätzlich ausfindig gemacht werden, welches Auto, auf welcher Lane zugeordnet wurde. Die Lane wird erst dann wieder freigegeben, wenn das Fahrzeug von oben bis unten durchgefahren ist. Der Patcher bekommt von außerhalb ein Signal, welches die Funktion *freeLane* triggert, um den State der Fahrbahn wieder auf '0' zu initialisieren. *CarPositioning* ist allerdings, auch für die Position innerhalb der Fahrbahn zuständig. Nachdem die Position des Autos festgelegt wurde, muss die Geschwindigkeit des Autos, in der es von oben nach unten animiert fährt, definiert werden. Dafür wurde ein zusätzlicher Patcher *moving* initiiert, der in einem bestimmten Spektrum, zufällige Geschwindigkeiten generiert. Da nun alle Bedingungen geschaffen wurden, kann infolgedessen die *CarEntity* getriggert werden und daraufhin auch der *SpaceController*. Im *SpaceController*, werden die zeitlichen Abstände definiert, zu welchem Zeitpunkt ein Auto aktiviert wird. Der Grund liegt darin, dass die Autos nicht in zu kurzen Abständen freigegeben werden, damit der Fahrer die Möglichkeit hat auszuweichen.

Das Fahrzeug, welches der Fahrer steuert, wird wie schon oben beschrieben, als rotes Rechteck am unteren Rand des Fensters angezeigt. Hierfür wurde auch ein eigener Patcher *MainController* entwickelt, der für die Interaktion des Fahrers, mittels eines Controller, verantwortlich ist. Zu Beginn konnte das Fahrzeug, über das Keyboard des Rechners gesteuert werden. Da ein nahes Erlebnis in Bezug auf das Steuern eines echten Fahrzeuges für den Fahrer erzeugt werden sollte, wurde beschlossen ein Gaming-Lenkrad für die Simulation zu verwenden. Grundsätzlich kann ein Lenkrad an einen Rechner angeschlossen und dessen Daten in Max übertragen werden. Diese Daten werden dann über das '*hi*' Objekt verarbeitet, um diese im Patcher nutzbar zu machen. Jedoch stellte sich heraus, dass das gewählte Lenkrad nur einen USB 1.0 Anschluss aufwies. Da aktuelle macOS-Computer keine USB 1.0 Kommunikation unterstützen, musste eine Alternative gefunden werden, um das Lenkrad für den besagten Kontext verwenden zu können. Infolgedessen wurde das Lenkrad zunächst an eine externe Microsoft Surface Pro 3 [Sur] angeschlossen. Diese supportet die USB 1.0 Kommunikation und hat Max als integrierte Software. Auf der Surface befand sich der Patcher *ConnectorOSC*, der in Max ausgeführt wurde. Der *ConnectorOSC* hat die Signale des Lenkrades aufgenommen und verarbeitet. Darunter zählten Signale, ob das Lenkrad nach links, rechts oder nicht betätigt wurde. Außerdem wurde die Aktion der Blinker und eines Hebels für die Gangschaltung aufgenommen. Da eine Gangschaltung für die Simulation nicht vorgesehen war, wurde dieser für die Start-Stop-Funktion der Simulation umgewandelt. Für die Signalabfrage, wurde ein Zyklus

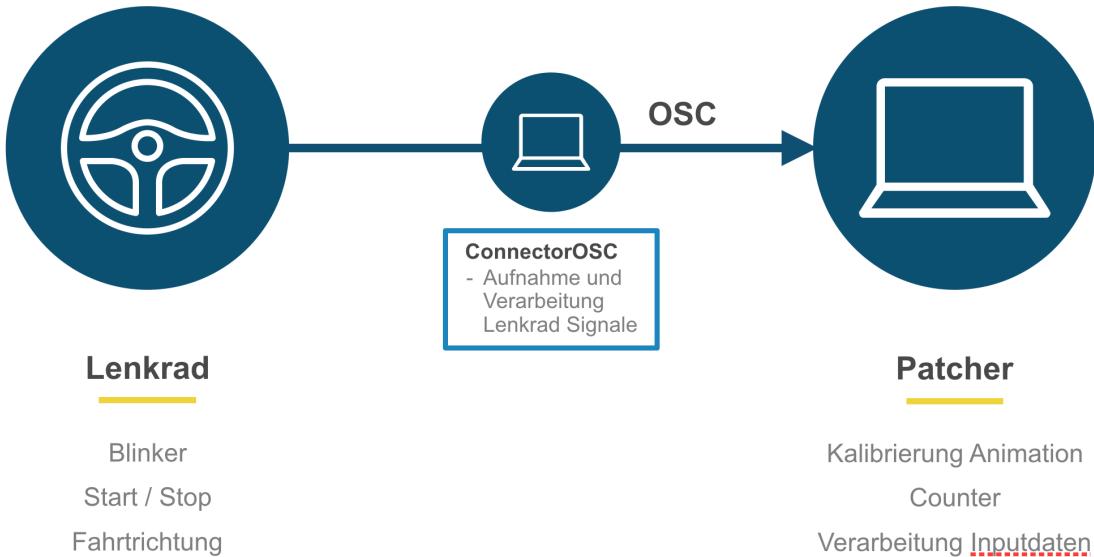


Abbildung 4.3: Darstellung der OSC-Übertragung

von 30 Millisekunden festgelegt, um ausreichende Daten des Lenkrades zu erhalten. Da eine hohe Dichte von Abfragen existiert, wurde ein '*AdaptiveLowpassFiltering*' hinzugefügt, der die Updatefrequenz reduziert, wenn keine Änderungen der Lenkradposition vorliegt. Die verarbeiteten Signale werden dann, über das OSC-Protokoll, an den Port 7005 am Rechner der Simulation gesendet. Der Patcher *OSCPatch* am Simulationsrechner, empfängt die Daten, um sie anschließend zu extrahieren. So können sie separat bearbeitet und benutzbar gemacht werden. Der *MainController*-Patcher hat nun Zugriff auf die Lenkraddaten und kann diese verarbeiten. Der *MainController* enthält zunächst die Abstraction *SteeringConverter*, welche die Sensitivität der Lenkwirkung definiert. Dabei war zu beachten, dass das Auto bei der kleinsten Bewegung nicht zu sehr ausschwenkt, um ein angenehmes und realitätsnahe Fahren zu generieren. Damit der Fahrer sehen kann, dass er einen Blinker betätigt hat, wurde ein weiterer Patcher *flashLight* hinzugefügt. Dieser beinhaltet ein gelbes *gridshape*, welches abhängig von der Betätigung des Blinkers, für die Anzeige in einem bestimmten Zeitfenster, ein und aus geschalten wird. Somit entsteht ein Blinken, welches in den oberen Ecken des Fahrzeuges platziert wird, wie in 4.1 zu sehen.

Der *MainController* enthält noch weitere Patcher. Der Patcher *FlashVerifier* beispielsweise, ist für die Prüfung der korrekten Blinkerbetätigung zuständig. Der *FlashVerifier* enthält ein Skript, welches verschiedene Zustände des Autos misst und bei Übereinstimmung Messages auslöst. Wenn der Fahrer über einen Fahrstreifen fährt, ohne zu Blinken, wird die Message *omittedFlash* ausgelöst. Diese wird wiederum an einen



4 Prototyping

anderen Patcher *FlashDetector* gesendet, welcher die Anzahl der vergessenen Blinker hoch zählt, wie in 4.2 dargestellt. Eine weitere Funktion *wrongDirection* gibt an, ob der Fahrer den Blinker zwar betätigt hat, jedoch in die entgegengesetzte Richtung gefahren ist. Zuletzt überprüft die Funktion *didFlash*, ob der Fahrer passend zur Fahrtrichtung geblinkt hat. Diese Parameter geben einen Einblick darüber, wie hoch die kognitive Auslastung der Tester, bei der Bewältigung der primären und sekundären Aufgabe in Kombination der tertiären ist.

Um den Fahrer und Moderator auf eine Kollision mit der Leitplanke oder einem anderen Auto aufmerksam zu machen, siehe 4.1, wurde zusätzlich ein Kollisionssound eingebaut. Eine *gridshape* weist keine genauen Kanten- und Linienlängen auf. Ihre Größe wird durch einen 3D-Skalierfaktor definiert, weshalb sich der Ursprung wiederum in der Mitte befindet. Jedoch werden Eckpunkte und Längen der jeweiligen *gridshapes* benötigt, um eine Kollision zu messen. Diese werden im Patcher *CollisionDetection* berechnet, wie in 4.2 zu sehen. Dieser führt zunächst, die Positionsdaten der Autos und des Fahrers zusammen, welche dann an ein Skript *CollisionDetection.js* übergeben werden. Dieses berechnet die Kantenlinien, der jeweiligen Autos vom Ursprung x aus. Wenn beide Autos aufeinander treffen wird eine '1' gesendet, die einen Collision Counter hoch zählt. Außerdem wird das Ergebnis, zurück an den *CollisionSound* gesendet, damit ein kurzes Signal ertönt.

Um spezielle Dinge wie die Geschwindigkeit, Start-Stop der Simulation und Counter zusammen zu führen, wurde ein MiraFrame hinzugefügt. Dieses Frame kann auf externen Geräten dargestellt werden. So können einfache und schnelle Änderungen realisiert werden. Dies ist hinsichtlich des Usability-Tests essentiell, um die Fahrsituation für den Kontext anzupassen.

4.1.2 Conversational Design

Um den Fahrer während der Transaktion zu unterstützen, wurde der Ansatz des multimodalen Conversational Interfaces angestrebt. Das bedeutet, dass der Fahrer zusätzlich zum Sprachassistenten, eine optionale grafische Unterstützung zu dem jeweiligen Themengebiet erhält. Die im Zuge der Arbeit bereitgestellte Prototyping-Umgebung der Firma adorsys GmbH & Co. KG wurde verwendet, um das Testen des Prototypen zu erstellen[ado]. Das Prototyping eines CUI ist im Gegensatz zu einem GUI dementsprechend aufwendiger, da das System auf spontane Reaktionen des Users eingehen muss.



4 Prototyping

Um nun die Gebrauchstauglichkeit der vorgestellten Einheit, mit natürlicher Sprache zu testen, wurde die von J. F. Kelley entwickelte Wizard-of-Oz-Methode verwendet. Diese sieht vor, Technologien schon in einem frühen Entwicklungsstadium zu testen, um Fehler in der Implementierung vorzubeugen. Das Verfahren involviert hierbei eine Person als Operator, welche die Antworten eines Systems simuliert, wenn der Nutzer mit der Bedienoberfläche agiert. Bei solchen Tests, wird meist nicht erwähnt, dass ein Mensch hinter der Logik der Interaktion steckt, um dem Nutzer das Gefühl eines voll funktionierendem Systems zu geben (vgl. [Wiz]). Mithilfe dieser Methode, kann eine reibungslose Konversation und somit quasi eine reale Situation wie der User mit dem System interagieren würde, generiert werden. Der Tester kann durch die Methode ausfindig machen, an welchen Stellen bei der Interaktion mit dem System Schwierigkeiten entstehen könnten.

Umsetzung

Damit eine Konversation mit dem User geführt werden kann, muss zunächst ein Szenario bzw. eine Dialogstruktur überdacht werden, um auf die Antworten des Users reagieren zu können. Da Nutzerinput grundsätzlich nicht ignoriert werden sollte, kann die aufgabenorientierte Konversation, in eine vorgesehene Richtung gebracht werden. Die Struktur kann hierbei über ein Storyboard erfolgen, wie in A.1 zu sehen ist. Da es sich um ein multimodales Conversational Interface handelt, wird nicht nur gesprochener Text, sondern auch grafische Elemente, wiedergegeben. Daraus ergibt sich die Folge, dass Mock-Up-Screens benötigt werden, um einzelne Inhalte bezogen auf gesprochene Phrasen, bildlich zu unterstützen. Es war zu beachten, dass die Gestaltung solcher Mock-Up-Screen nur dezidiert Informationen aufzeigen und nicht mehr Auskunft geben, als vom Sprachassistenten wiedergegeben wurden. Dies kann über wenig Text und nachvollziehbaren Gestaltungselementen erreicht werden. Da das Interface auch über manuellen Input ansteuerbar sein sollte, muss die Gestaltung für diesen Aspekt, ausgelegt werden. So kann das Interface als grafische Unterstützung für den Fahrer verwendet werden. Alle Mock-Up-Screens werden im Anhang ausführlich aufgeführt, siehe A.2.

Die Gestaltung der Mock-Up-Screens, hat sich hierbei auf das Szenario bei der Zahlung eines 10-Tages-Mautticket bezogen, welches für die Einreise nach Österreich benötigt wird. In Bezug dazu, wird beim Kaufvorgang des Mauttickets, abhängig vom Land in welches der User einreist, der Name und die Silhouette des jeweiligen Landes



4 Prototyping

angezeigt. Die Screens ziehen sich als Konzept durch den Einkaufsprozess.

Um auch für alternative Konversationsverläufe gesichert zu sein, wurden unterschiedliche Storyboards mit zusätzlichen *Mock-Up-Screens* gestaltet, um flexibel auf die Antworten des Users, eingehen zu können.

Um das Szenario für die Wizard-of-Oz-Methode umsetzen zu können, wurde ein *Conversation Prototyping Tool* (CPT) verwendet, welches wiederum in Max erstellt wurde¹. Für das CPT existieren zunächst ein *Operator-Patcher* und *Tester-Patcher*, auch Panels genannt. Beide werden als Bedienoberfläche für die jeweiligen Personen verwendet, um mit dem System zu interagieren. Der *Operator-Panel* befindet sich folglich auf dem Rechner des Operator, der für die Steuerung der Antworten, bezogen auf die Reaktionen des Users, zuständig ist. Der *Tester-Panel* zeigt die jeweiligen Antworten des Operators an, welche als grafische Elemente und als Audiodateien, wieder gegeben werden, siehe 4.4. Damit beide Rechner kommunizieren können, sind diese in einem Netzwerk miteinander verbunden. Die Daten des Operators werden über das *User Datagram Protocol* (UDP) vom *Operator-Rechner*, zum *Tester-Rechner* mit entsprechenden Ports geschickt. Um auf die Funktionen des *Operator-Panels* näher einzugehen, ist zu beachten, dass die Benutzeroberfläche in Bezug eines vordefiniertem Szenario, dynamisch generiert werden kann. Hierbei können auf Audiophrasen und UI-Elemente zurückgegriffen werden, um Daten an den *Tester-Panel* zu schicken. Diese Elemente bzw. Bedienoberfläche baut auf einer JSON-Datenstruktur auf, welches individuell angepasst werden kann.

```
1 {
2     "name": "PricePicker1Intent",
3     "priority": {
4         "main": 0
5     },
6     "utterances": [
7         "Ja ich moechte das Ticket erwerben.",
8         "Ja ich moechte ein Ticket fuer 10 Tage bitte.",
9         "Wie viel kostet das Ticket?"
10    ],
11    "responses": [{{
12        "subject": "Price10Days",
13        "phrase": "Dieses Mautticket hat einen Preis von 9,40 Euro.
14            Moechtest du es erwerben?",
```

¹Die Arbeit des CPT stammt vom CUI-Team der adorsys GmbH & Co. KG unter der Leitung von Steffen Blümm



4 Prototyping

```
15      "modality": "voicefirst",
16      "tags": ["price", "austria", "pickerl"],
17      "payload": {
18          "durationAmount": "10",
19          "durationUnit": "Tage",
20          "price": "9,40"
21      },
22      "audiofile": "Pickerl_TimePeriod_PickerlPriceDays.aiff"
23  }
```

Die *phrase* gibt an, welche Audiophrase bei der Bedienung des *Intents* abgespielt wird. Es ist zu beachten, dass das jeweilige *Audiofile* durch ein *Phrase Render Tool* (PRT) generiert wurde. Dieses verwendet die *Text-to-Speech* (TTS) Technologie, welches geschriebenen Text in Audio-Dateien im *Audio Interchange File Format* (AIFF) rendert. Für das angegebene Szenario, wurde hierbei das PRT der adorsys verwendet, um eine JSON-Datenstruktur in eine Audiodatei zu wandeln. Um der Antwort nun auch ein UI - Element mitzugeben, werden bei *type* und *tags* entsprechende Werte zugewiesen. Mit der *payload* können dann zusätzliche Werte definiert werden, die dynamisch für das jeweilige Szenario geändert werden können. Dies gilt auch für die Werte des Picture Passcodes. Die Darstellung der Operator-Bedienoberfläche, wie in 4.4 zu sehen, setzt sich aus Tabs zusammen, die abhängig des beschriebenen JSON angelegt werden. Bei der Gestaltung des JSON, wurde darauf geachtet, dass dieses dem Aufbau einer Konversationsstruktur ähnelt. In der Regel wird ein JSON-Szenario mit mehreren *Intents* aufgebaut, um auf jeweiligen Absichten des Nutzers einzugehen. Jedem einzelnen *Intent* werden mehrere Antworten zugewiesen, auch *Utterances* genannt.

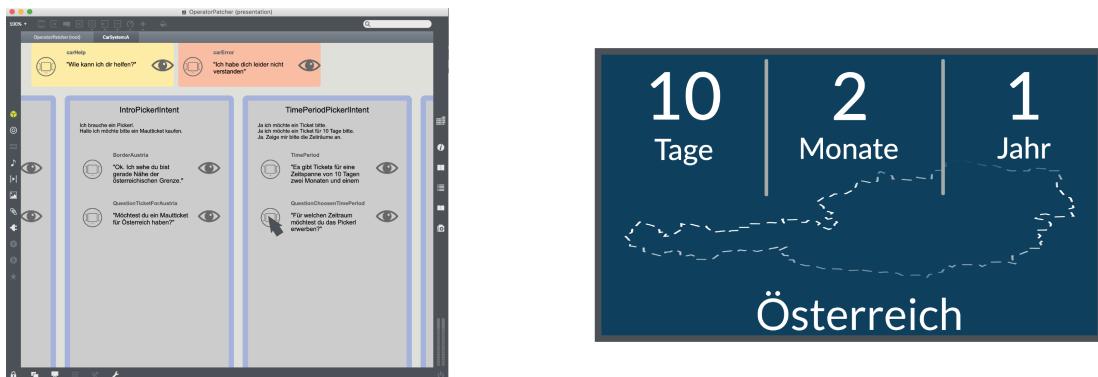


Abbildung 4.4: Links: Operator-Panel, Rechts: Jeweiliger Tester-Panel



4 Prototyping

Diese sind als Textbausteine in den Tabs der *Intents* dargestellt. Die UI - Elemente und deren jeweiligen Phrasen, werden an den *Tester*-Panel des Probanden geschickt. Hier werden zwei Patcher verwendet. Der *Display*-Patcher ist für die Darstellung der Screens zuständig und der *Setup*-Patcher für die Ausgabe der jeweiligen Audiodateien. Mit diesen Voraussetzungen kann nun ein Usability-Test implementiert werden.

4.2 Usability-Test

Bei einem Usability-Test interagiert der repräsentative Benutzer in der Regel mit einem interaktiven System. Es sollen Usability-Probleme des Systems identifiziert und analysiert werden, um falsche Annahmen für das System zu vermeiden. Es wird gezielt auf das Verständnis der Erfordernisse an ein System examiniert und auf das Verhalten des Benutzers bei der Interaktion fokussiert. Für den besagten Kontext werden qualitative Usability-Tests durchgeführt und hierbei auf die *Think-aloud*-Methode zurückgegriffen (vgl. [Hop18]). Die Testperson formuliert ihre Gedanken laut, sodass die Moderatoren Rückschlüsse auf die Interaktion des Users machen können.

In Bezug dazu, wurde zunächst die Funktionalität des Picture Passcodes als geeigneter Autorisierungsprozess im Kontext Fahren getestet. Der Fokus der Tests lag hierbei auf den ergonomischen Aspekten. Darunter zählte die Messung des Task-Loads für den Fahrer. Dieser wurde durch Messpunkte definiert, wie beispielsweise die Anzahl der Kollisionen. Außerdem wurde untersucht, in welcher Zeit der User zur Erfüllung der tertiären Aufgabe benötigt und ob er sie erfüllen konnte. Als letzter Testaspekt wurde der Nasa TLX verwendet, zur Selbsteinschätzung des Users.

Für die vorliegende Arbeit wurden zunächst zwei Usability-Tests durchgeführt. Bei dem ersten Test handelte es sich um Pre-Tests. Diese waren notwendig, um gewisse Arbeitsabläufe einzustudieren, weitere Verbesserungsvorschläge aufzunehmen und umzusetzen. Außerdem konnten dadurch Fragestellungen in Bezug auf die Test-Sessions überarbeitet werden, um gezieltere Erkenntnisse der Eindrücke zu gewährleisten. Der zweite Usability-Test wurde als offiziell eingestuft und übernahm die Verbesserungen und Abläufe der vorherigen Pre-Tests.

Für beide Tests waren von der Firma adorsys GmbH & Co. KG, der Technical Lead von CUI Steffen Blümm, als Leiter der Tests, der Verfasser dieser Arbeit als Moderator



4 Prototyping

und die Werkstudentin Mona Kögel als Operator anwesend. Der Moderator führte den User durch den Test und eröffnete anschließend das Interview und die Diskussion.

4.2.1 Testaufbau

Bevor ein Usability-Test gestartet werden konnte, mussten vorweg einige Vorbereitungen getroffen werden. Hierbei war zunächst ein geeignetes Szenario zu wählen, welches an eine realistische Verwendung des Picture Passcodes, anlehnt. Die detaillierte Version ist im Anhang zu finden A.3.

Szenario

Beschreibung: Du fährst mit deinem Auto nach Österreich. Aus diesem Grund möchtest du dir ein Maut Ticket für 10 Tage kaufen. Das Ticket kaufst du während du fährst über den Sprachassistenten in deinem Fahrzeug

Aufgabenstellung: Nutze den Sprachassistenten um ein Mautticket für 10 Tage zu kaufen.

Deine Secrets:



Da eine Registrierung im Prototypen und für den Usability-Test nicht vorgesehen war, wurden dem User die *Secrets* vorgegeben. Zusätzlich dazu konnte untersucht werden, ob der User den richtigen Passcode sagt und somit die Aufgabe erfolgreich erfüllen konnte. Das Szenario wurde für die gesamten Tests angewendet. Um nun Unterschiede und Vergleiche für bestimmte Use-Cases machen zu können, wurde ein A-und B-Test konzipiert. Für die Pre-Tests wurde evaluiert, welchen Einfluss die Position des Displays auf das Fahrverhalten des Probanden hat.



4 Prototyping

Für den A-Test wurde der Rechner neben das Lenkrad gestellt, um das Display in der Mittelkonsole zu imitieren. Für den B-Test wurde das Interface hinter dem Lenkrad platziert, um ein Dashboard-Display darzustellen. Da sich durch die Pre-Test, kein signifikanter Einfluss der Position des Displays auf den Task-Load herausstellte, wurde der A/B-Test nochmals unter einen weiteren A- und B- Test geteilt. Diese Tests geben Aufschluss darüber, wie sehr sich die Variation der Geschwindigkeit, auf die Beanspruchung des Fahrers auswirkt.

Für einen A/B-Test wurden zwei JSON-Szenarios generiert, die sich durch unterschiedliche Picture Passcodes unterscheiden. Falls ein Proband, eine falsche Angabe des Passcodes macht, erhält er einen weiteren Versuch. Wenn die Angabe des Passcode allerdings zweimal inkorrekt war, wurde der Usability-Test abgebrochen und als nicht erfolgreich klassifiziert.

```
1 "payload": {  
2     "firstIconNames": [  
3         "sydney",  
4         "NYC",  
5         "rom",  
6         "rio",  
7         "sanFrancisco",  
8         "berlin",  
9         "barcelona",  
10        "london"  
11    ],  
12    "firstNumbers": [1, 3, 4, 12, 41, 31, 2, 13],
```

Für die Wahl der Testpersonen in Bezug der Pre-Tests, wurden zunächst zwei Teilnehmer gesucht, welche gewissermaßen Vorkenntnisse über das System mitbrachten. Nach der Absolvierung der Tests, wurden diese über die Selbsteinschätzung in Bezug der Fahrerpraxis, Multitaskingfähigkeit, Ablenkbarkeit und Praxis mit der Verwendung eines Sprachassistenten, befragt. Für die offiziellen Usability-Tests, wurde auch eine Selbsteinschätzung der Tester durchgeführt. Allerdings fand diese mittels einer Online-Umfrage, vor den eigentlichen Tests statt. Der Grund liegt darin, nach diesen Kriterien vergleichbare Gruppen für die A/B-Tests zu generieren. Dabei wurden 12 Teilnehmer ausgewählt und paarweise zusammengestellt, die ähnliche Angaben bei der Selbsteinschätzung machten. Somit ergaben sich zwei Gruppen, jeweils sechs Personen, die entweder das Display in der Mittelkonsole oder Dashboard erhalten haben.

4 Prototyping

Jeder Teilnehmer bekam den Auftrag, die Aufgabenstellung bei einer flotten und gemütlichen Geschwindigkeit der Simulation, zu absolvieren. Lediglich die Reihenfolge dessen, hat sich bei den Teilnehmern unterschieden. Bei der Wahl der Probanden wurde zudem auf die gleichmäßige Verteilung in Bezug des Geschlechtes, Alter und der Selbsteinschätzung geachtet. Die Selbsteinschätzung hat allerdings den größten Teil bei der Entscheidung eingenommen. Die komplette Übersicht der Teilnehmer kann im A.7 entnommen werden.

Der prinzipielle Aufbau des Usability-Tests, ist in 4.5 zu sehen. Dieser setzt aus der prototypischen Fahrsimulation 4.1 und dem beschriebenen Conversational Design 4.1.2 zusammen.



Abbildung 4.5: Testaufbau der Usability-Test

Für den Usability-Test wurde ein großer Raum gewählt, sodass der Operator nicht auffällig in Erscheinung getreten ist. Außerdem wurde der Rechner für die Fahrsimulation über HDMI mit einem Fernseher verbunden, um eine realistische Breite der Fahrerperspektive auf die Strasse zu ermöglichen.

Testablauf

Bevor der eigentliche Test begann, musste zunächst jeder Proband, eine Zustimmung zur anonymen Verarbeitung der Daten unterschreiben, siehe . Anschließend wurde der Person das Szenario, den Picture Passcode mittels eines Videos und Funktionsweise



4 Prototyping

des Lenkrades näher erläutert. Damit der User etwas Gefühl für die Simulation bekam, konnte dieser eine kurze Testfahrt machen bevor er mit der eigentlichen Aufgabe beginnen musste. Nachdem die Tests beendet waren, folgte für jeweils beide der Nasa TLX, der im folgenden Abschnitt 4.2.2 näher beleuchtet wird. Zuletzt wurden zusätzliche Fragen des Moderators und des Leiters an den User gestellt.

4.2.2 Nasa TLX

Der *Nasa TLX* (Task-Load-Index) ist eine subjektives Verfahren zur Messung der Beanspruchung des Fahrers. Die Beanspruchung wird in sechs Dimensionen geteilt:

- Mental Demand: mentale Beanspruchung
- Physical Demand: körperliche Beanspruchung
- Temporal Demand: zeitliche Beanspruchung
- Performance: Einschätzung der eigenen Leistung
- Effort: eigener Aufwand
- Frustration: Frustrationslevel bei Bearbeitung

Jede Dimension wird in einer Skala mit 21 Abstufungen angegeben und vom User beurteilt. Der Proband entscheidet darüber, welche Dimension im Vergleich zu einer anderen Dimension wichtiger erscheint, sodass eine Gewichtung gebildet wird (vgl. [HS88, S. 178]). Im Usability-Test folgte der Nasa TLX direkt nach der Abschließung des A/B-Tests. Die Gewichtung der Dimensionen für beide Tests ergab sich, nachdem der letzte Test ausgeführt wurde. Für den Usability-Test wurde eine selbstentwickelte Nasa TLX Applikation eines iOS Gerätes verwendet.

4.2.3 Evaluation

Im folgenden Abschnitt werden die Ergebnisse der Usability-Tests zusammengetragen und evaluiert. Zunächst werden die Messwerte Geschwindigkeit, Erfolgsrate und Spurwechselverhalten (Blinken) berücksichtigt. Im Anschluss werden die Ergebnisse des *Nasa TLX* für die A-/B-Tests ausgewertet. Darauffolgend wird die Evaluation der Fragerunde aus den Gesprächen mit den Probanden aufbereitet.

4 Prototyping

Messwerte Usability-Tests						
	Erfolgsrate	Dauer	Collision Count	Lane Crossing Correct	Omitted	Wrong Direction
Insgesamt mit Pre-Tests	88 %	20s	2	92,3 %	6,9 %	0,8 %
Insgesamt ohne Pre-Tests	81 %	17s	2	90,0 %	9,0 %	1,0 %
Pre-Tests	100 %	20s	0	98,4 %	1,6 %	0 %
Flott	75 %	25s	2	96,5 %	3,5 %	0 %
Gemütlich	80 %	12s 500ms	0	83,5 %	14,5 %	2,0 %

Abbildung 4.6: Messwerte der Usability-Tests

Aufgrund bestimmter Umstände, konnten nicht alle Tests durchgeführt werden. Aus diesem Grund, wurden zusätzlich die Messwerte der Pre-Tests berücksichtigt. Dabei hat sich eine 88% Erfolgsrate bei der Autorisierung über den Picture Passcode ergeben, wie in 4.6 zu sehen. Eine Person, konnte die Autorisierung im A-Test nicht erfolgreich abschließen, da die Funktionsweise des Picture Passcode nicht direkt klar war. Zudem kamen die zwei Kollisionen im A-Test von der selben Person zustande. Insgesamt kam es zu keinen weiteren Kollisionen. Dies deutet auf eine keine Überlastung der Probanden hin. Die Median-Dauer zur Lösung des Picture Passcode betrug 20 Sekunden. Somit werden die Bedingungen, die in 2.4.1 beschrieben sind, knapp erfüllt. Der höchste Wert lag für die Autorisierung bei 49 Sekunden der flotten Geschwindigkeit. Die niedrigste Dauer betrug 6 Sekunden in Bezug der gemütlichen Geschwindigkeit. Allerdings lässt sich nicht aussagen, ob die Geschwindigkeit der ausschlaggebende Aspekt zur Minderung der Dauer ist. Da drei Personen eine flotte Geschwindigkeit für den ersten Test erhielten, kann die Autorisierungsdauer abhängig der Vertrautheit mit dem System ausschlaggebend sein. Zuletzt lag die korrekte Überquerung der Mittelstreifen bei 92,3%. Die falsche Richtungsanzeige des Blinkers wurde mit 0,8% und der vergessene Blinker mit 6,9% ausgemacht. Die dargestellten Ergebnisse legen nahe, dass trotz der Beanspruchung durch die tertiäre Tätigkeit weiterhin erfolgreich auf eine sekundäre Aufgabe geachtet werden kann. Auch wenn keine weitreichenden Aussagen möglich sind, scheinen die Probanden in der Situation nicht an der Grenze ihrer Fähigkeiten.

Nasa TLX Ergebnisse

In der Abbildung 4.7 werden sechs Dimensionen des Nasa TLX vorgestellt. Die dunkelblauen Säulen bilden die Analyse des A-Tests (gemütliche Geschwindigkeit des Fahrers) und die hellblauen Säulen stehen für die Ergebnisse des B-Tests (flotte Geschwindigkeit des Fahrers).

4 Prototyping

Eine genauere Darstellung der Ergebnisse für jeden einzelnen User ist im Anhang A.2 und A.1 zu finden.

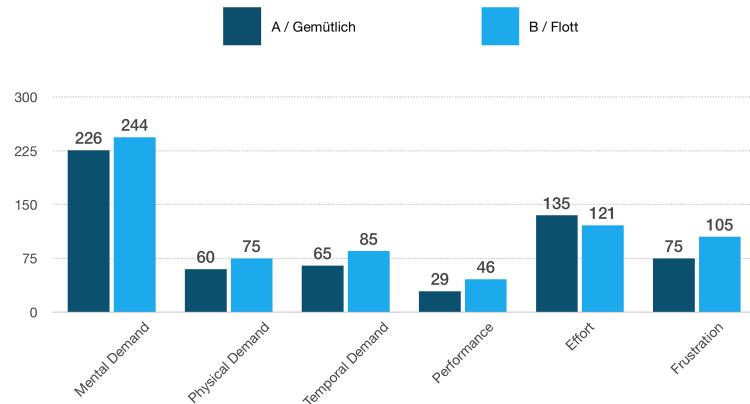


Abbildung 4.7: Ergebnisse der NASA TLX des Usability-Tests (eigene Darstellung)

Bei der Analyse der Ergebnisse ist zunächst zu beachten, dass nur vier der zwölf Personen bedingt der Corona-Krise, getestet werden konnten. Trotz dessen ließen sich einige Tendenzen, auch in Bezug der Pre-Tests, feststellen. Es fällt zunächst auf, dass der *Mental Demand*, also die mentale Beanspruchung, für alle Tester bei der Aufgabenbewältigung sehr hoch war. Dass eine flotte Geschwindigkeit eine höhere mentale Beanspruchung verlangt, ist darauf zurück zu führen, dass sich mehr auf den Straßenverkehr fokussiert wird. Außerdem bekamen drei Tester eine flotte Geschwindigkeit als ersten Test zugewiesen. Dabei stellte sich heraus, dass die Mehrzahl die Aufgabe beim zweiten Test als einfacher empfunden, da sie schon erste Erfahrungen sammeln konnten. Im Vergleich dazu, konnte die körperliche Beanspruchung, einen niedrigen Wert bei insgesamt drei Probanden aufweisen. Bei einer Person war dieser Wert, wiederum sehr hoch, wie im Anhang zu sehen A.1, da die Position des Bildschirmes zur einer längeren Blickabwendung zur Straßensituation forderte. Für den *Temporal Demand* lässt sich keine Prägnanz für einzelne Probanden feststellen. Allerdings stellte sich durch Gespräche heraus, dass ein gewisser Zeitdruck vorhanden war. Dieser wurde jedoch nicht durch die Methodik des Passcodes verursacht, sondern der selbstgemachte Druck, den Passcode in einer bestimmten Zeit finden zu müssen. Die Selbsteinschätzung der *Performance* zur Aufgabenbewältigung, haben die Probanden als hoch eingeschüftet, wie in 4.7 zu betrachten. Bezogen auf den *Effort* lässt sich durch die Auswertung und den Gesprächen herausfiltern, dass der Aufwand für die Aufgabe verhältnismäßig hoch war, da das Suchen der *Secrets* viel Aufwand mit sich brachte.



4 Prototyping

Zuletzt war die Frustration für die Probanden ausschlaggebend, wenn diese ihre *Secrets* nicht gefunden haben oder einen falschen Passcode angaben. Bezogen auf die Position des Lenkrades, konnte mit der geringen Anzahl an Probanden, keine prägnante Unterscheidung in der Performance gemacht werden, weshalb dies als Analyse nicht besprochen wurde.

Feedback aus den Gesprächen

Aus den Gesprächen mit den Probanden, wurde festgehalten, dass der Passcode als eine sichere Methode angesehen wird, um eine Bezahlung durchzuführen. Einige würden dieses Verfahren verwenden, wenn sich Familienmitglieder oder enge Bekannte im Auto befinden. Bei einer weniger bekannten Person, sind viele der Probanden skeptisch, weshalb sie von der Verwendung des Autorisierungsverfahrens, eher absehen. Viele der Testpersonen würden Sachen kaufen, die einen Bezug zur Fahrt haben, wie beispielsweise die Maut, Drive-In oder Benzin. Außerdem wären sie bereit kleinere Beträge oder dringende Rechnungen mit der Methode zu bezahlen. Sobald es sich um größere Beträge handelt, wird eher weniger dazu tendiert. Bezogen auf die Fahrsituation, würde die Benutzung des Passcodes während einer Autobahnfahrt eher stattfinden, als im Stadtverkehr. Auf der Autobahn herrschen in der Regel weniger Unterbrechungen und Richtungswechsel. Einige Probanden würden die erstmalige Benutzung des Autorisierungsverfahrens vor Antritt einer Fahrt ausprobieren. vor der ersten Benutzung im Stehen testen oder alleinig im Stehen verrichten. Andere würden die Funktion wiederum wertschätzen, wenn keine direkten Voreinstellungen gemacht werden müssten und das System sofort verwendbar wäre. Die Autorisierung per Picture Passcode lässt sich mit der allgemeinen Beanspruchung bei der Bedienung eines Navigationssystems vergleichen. Wie im oberen Teil beschrieben, war die Handhabung für den Großteil zudem leichter, wenn die Aufgabe schon einmal erledigt wurde. Das impliziert, dass die Performance durch eine öftere Verwendung gesteigert werden kann. In Bezug auf die Gestaltung des Picture Passcode, stellte sich heraus, dass die Icons für viele schwer zu finden und identifizieren waren. Grund dafür ist der schnelle Wechsel der Icon-Layer und die Ähnlichkeit einiger Icons.

Lernerfahrungen und Verbesserungsvorschläge

Die Tests gaben Aufschluss darüber, welche Verbesserungen für das System angebracht wären und welche Lernerfahrungen daraus geschlossen werden konnten.



4 Prototyping

Dabei kann der Passcode als eine nachvollziehbare Methode, um seine Autorisierungsdaten zu schützen, angesehen werden. Allerdings müsse die Dauer für die Bewältigung des Autorisierungsprozess gekürzt werden, um unter die vorgeschriebenen 20 Sekunden 2.4.1 zu fallen. Dabei hat sich die Größe und Gestaltung der Icons als wichtiger Punkt herausgestellt. Um die Dauer für die Suche der *Secrets* zu kürzen, besteht die Möglichkeit diese zumal zu vergrößern oder markantere Motive zu wählen. Der Wechsel der Layer, erwies sich für einige Probanden, als zu schnell. Allerdings wäre es weniger sinnvoll die Anzeigedauer der jeweiligen Icon-Layer zu verlängern. Da der Fahrer nur kurze Blicke für die Erkennung verwenden kann, wäre bei einem längeren Wechsel die Gefahr, dass der Fahrer bei jeder Blickabwendung, zufällig auf die gleichen Layer blickt. Bei einem sekündlichen Wechsel kann der Fahrer, einen kurzen Blick, innerhalb von weniger als 2 Sekunden, auf zwei Layer erhalten. Außerdem erkennt der Fahrer beim ersten Blick, ob es sich lohnt diesen kurz verweilen zu lassen bis der nächste Layer eingebendet wird. Womöglich hat der Mangel an Fahrpraxis und Multitaskingfähigkeit auch einen Einfluss darauf, ob tertiäre Aufgabe eher weniger wahrgenommen wird, da sich mehr auf das Fahren fokussiert wird. Für einige Probanden hat die Sicherheit in Bezug der Daten auch eine Rolle gespielt, da sie das Risiko für höhere Beträge nicht eingehen würden. Hierfür wurde vorgeschlagen, einen dritten Icon-Layer hinzuzufügen, um eine niedrigere Wahrscheinlichkeit für das Erraten der *Secrets* zu erzielen.

4.3 Funktionale Komponenten

Damit der Picture Passcode für ein System implementiert werden kann, müssen Bedingungen in Bezug der Anzeige und Logik der Zahlengenerierung, geschaffen werden. Es wird zunächst auf die Logik der Anzeige des Picture Passcode eingegangen. Für den Usability-Test wurde zunächst ein prototypischer Ansatz gewählt.

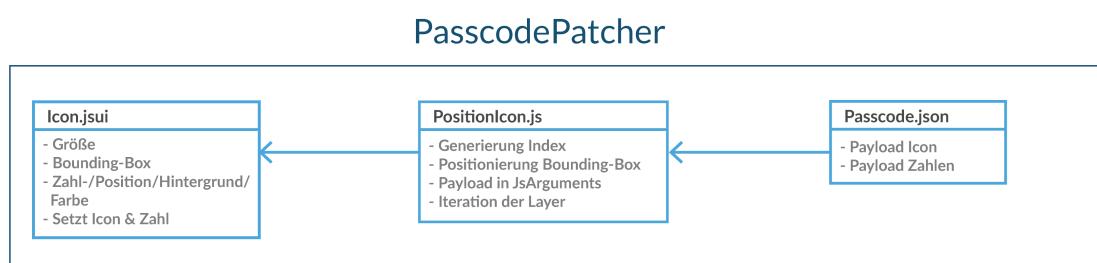


Abbildung 4.8: Funktionsweise den Prototypen für den Picture Passcode



4 Prototyping

Die Eigenschaften der Icons, wurde über ein *jsui*-Objekt [Jsu] definiert. Dieses beschreibt zunächst die Größe des Icons und dessen Bounding-Box, um ein Raster bilden zu können. Außerdem wird die Position, der Hintergrund und die Farbe der Zahl bestimmt, die über das Icon gelegt wird, siehe 4.8. Die Faktoren, welche Zahl zu welchem Icon und welcher Reihenfolge gehört, werden abhängig von den Daten bzw. Argumenten gemacht, die über ein JSON-Skript (*Passcode.json*) mitgegeben werden. Dabei erhält jedes Icon-Display einen Index, der definiert, an welcher Position das Icon im Raster, platziert wird. Den Index erhält das *Icon.jsui* über Argumente, die in einem anderen js-Skript (*PositionIcon.js*) festgelegt werden. Das *PositionIcon.js* gibt die Position der Icon-Boxen vor, um ein Setzkastensystem zu implementieren. Da die Anzahl der vorgegeben acht Icon-Boxen für eine 2 x 4 Matrix vorgeschrieben sind, könnte ein Skript geschrieben werden, welches die Größe der Matrix, über die Anzahl der Icons und die Größe des Displays, skalierbar macht. Damit nun unabhängig und dynamisch ein Passcode generiert werden kann, muss ein Algorithmus geschaffen werden, der bestimmte Funktionen für dieses Vorgehen implementiert. In 3.3.2 wurden hierbei Bedingungen definiert, die der Picture Passcode erfüllen muss und in einem Skript umgesetzt. Zu Beginn wird eine dreistellige Zahl generiert, welche darauf geprüft wird, dass sie keine 0 enthält. Außerdem wird mit dem Skript sichergestellt, dass der Passcode keine dreistellige auf- oder absteigende Zahlenfolge beinhaltet. Zuletzt werden Repdigits exkludiert. Das Skript ist im Anhang A.9 zu entnehmen.

Da aus zeitlichen Gründen an der Implementierung nicht weitergearbeitet wurde, konnte jedoch eine konzeptionelle Beschreibung ermöglicht werden. Da der Passcode nun definiert wurde, muss zunächst ein Zahlenspektrum bestimmt werden. Dieses setzt sich aus den drei Ziffern des Passcodes zusammen. Im Anschluss wird eine weitere Zahl definiert, um die Menge zu komplementieren. Der Passcode kann aus zwei verschiedenen Zahlkombinationen zusammengesetzt werden, weshalb zwei zweistellige Zahlen festgelegt werden können. Da jede einstellige Zahl doppelt vorkommt, konnten zehn Zahlen erstellt werden. Somit werden noch sechs weitere Zahlen, für den Autorisierungsprozess benötigt. Diese können aus einer Menge von 14 Zahlen definiert werden (mögliche zweistellige Zahlenkombinationen mit den vordefinierten vier Ziffern). Bei der Wahl sollte jedoch auf eine gleichmäßige Verteilung der Ziffern geachtet werden, sodass nicht ausschließlich zweistellige Zahlen einer Zahlenfolge gewählt werden. Mit anderen Worten, die Auswahl sollte bei einer Beispielmenge [1, 4, 6, 9] nicht ausschließlich auf die Zahlenkombinationen 41, 44, 46, oder 49 fallen, um ein ausgeglichenes Verhältnis zu erhalten. Nachdem 16 Zahlen festgelegt wurden, können



4 Prototyping

diese zu den jeweiligen Icons bzw. *Secrets* gesetzt werden. Es muss die Bedingung geschaffen werden, dass jeder einstelligen Zahl, eine zweistellige Zahl zugewiesen wird, die im späteren Verlauf eine dreistellige Zahl ergeben. Die Verteilung der Zahlen, zu den jeweiligen Icon-Boxen sowie die Positionierung bzw. Festlegung des Index der jeweiligen Icon-Boxen, erfolgt zufällig. Ausschließlich für den Passcode gilt die Besonderheit, dass die jeweiligen Icon-Boxen nicht den gleichen Index aufweisen dürfen, damit die *Secrets* nicht auf der gleichen Position liegen. Falls die Autorisierung beim ersten Versuch nicht erfolgreich war, muss ein neuer Passcode generiert werden. Es muss darauf geachtet werden, dass sich der Passcode zum vorherigen Picture Passcode in Bezug des Zahlenspektrums und der Position der *Secrets* prägnant unterscheiden.



5 Schlussbetrachtung

Das letzte Kapitel fasst die gesamte Arbeit kurz zusammen, bewertet diese und gibt einen Ausblick auf zukünftige Weiterentwicklungsmöglichkeiten des vorgestellten Konzeptes.

5.1 Zusammenfassung

Zahlen im Auto, ist in den letzten Jahren kontinuierlich in den Vordergrund gerückt, weshalb viele Unternehmen verstärkt Lösungen suchen, um dieses Bedürfnis zu erfüllen. Da der Autorisierungsschritt bei vielen vorgestellten Systemen vernachlässigt wurde, war das Ziel dieser Arbeit ein Autorisierungskonzept für die Initiierung von Bezahlungen zu entwickeln, welches auch während der Fahrt nutzbar ist. Dabei wurde zunächst die Methodik grafischer Passwörter betrachtet, um im späteren Verlauf ein Konzept zu entwickeln, welches auf Basis des *Picture Password* von Blackberry, aufbaut. In Bezug dazu, wurden einige Änderungen vorgenommen, um dieses ebenfalls über die Verwendung eines multimodalen Conversational Interface zu garantieren. Das Resultat ist ein dreistelliger Zahlencode, auch Picture Passcode genannt. Dieser wird über die Zusammensetzung zweier Zahlen bestimmt, die auf ausgewählten Secrets bzw. Icons des Users erscheinen. Um die Funktionalität des Picture Passcodes, in Bezug der Verwendung während der Fahrt zu testen, wurde zunächst ein Usability-Test angedacht. Die geeigneten Bedingungen wurden durch eine Fahrsimulation geschaffen, welche mentale, körperliche und zeitliche Beanspruchung bei der Steuerung eines Fahrzeuges imitiert. Zudem wurde die prototypische Implementierung eines multimodalen Conversational Interface generiert, welche mittels der Wizard-Of-Oz Methode getestet wurde. Ob sich der Picture Passcode als eine geeignete Autorisierungsmethode eignen könnte, wurde über Messwerte und der eigenen Einschätzung der User des Nasa TLX evaluiert.



5 Schlussbetrachtung

Schlussendlich wurden noch einige funktionale Komponenten vorgestellt, um die Methodik des Picture Passcode, für ein reales System zu implementieren.

5.2 Fazit

Die Forschungsfrage der Arbeit konnte bisher nur bedingt beantwortet werden. Da bis dahin nur eine geringe Anzahl an Usability-Tests durchgeführt werden konnte, ist die Auswertung nur bedingt repräsentativ. Allerdings ließen sich trotz dessen einige Erkenntnisse gewinnen. Der Picture Passcode stellte sich als eine mögliche Autorisierungsmethode für den User, dar. Der Passcode wäre eine Option, um kleine, dringende oder fahrbezogene Beträge zu bezahlen, auch wenn sich eine bekannte Person im Auto befindet. Bezogen auf die Ergonomie, müssen allerdings einige designtechnische Aspekte verbessert werden, da das System bis daher nicht für den Straßenverkehr geeignet wäre. Außerdem ließe sich der Passcode bei einer schwierigen Verkehrssituation nicht bedienen, da dies eine lange Blickabwendung erfordert und andere Straßenverkehrsteilnehmer und die eigene Person in Gefahr bringen würde. Die Verwendung auf der Autobahn oder während einer Situation mit wenig Verkehrsaufkommen wäre allerdings auch für die Tester eine Option.

5.3 Ausblick

Die bisherigen Tests konnten einen vielversprechenden Ausblick in die weitere Forschung des Konzeptes einbringen. Aus diesem Grund wäre für die Weiterführung der Arbeit eine weitere Anzahl an Usability-Tests angedacht, um fundamentale und investigative Ergebnisse zu erhalten. Die Evaluation kann verwendet werden, um konzeptionelle und design-technische Aspekte des Picture Passcodes zu überarbeiten. Dadurch kann ein endgültiges Konzept implementiert werden, welches Verkehrs- und Daten sicherheitstechnische Faktoren für die Fahrt erfüllt. Da die Verwendung der Autorisierungsmethode auf der Autobahn in Frage käme, könnte der Picture Passcode, insbesondere in Verbindung mit Fahrassistentensystemen (Spurhalteassistent, Bremsassistent - Hochautomatisiertes Fahren), welche dem Fahrer bei den primären Fahraufgaben unterstützen, für Autobahn- und Schnellstrassenstrecken fortführend evaluiert werden.



5 Schlussbetrachtung

Zusätzlich, könnte in Bezug auf kontaktloses Zahlen in Pandemiezeiten, eine Verwendung am Point Of Sale untersucht werden. Zuletzt wäre eine Integration für eine professionelle Fahrsimulation zu implementieren, um Tests schaffen zu können, die noch näher an reale Verkehrssituation erinnern.

Danksagung

Abschließend bedanke ich mich für die Zusammenarbeit mit der Firma adorsys GmbH & Co. KG und insbesondere bei Steffen Blümm für die kompetente Betreuung und Unterstützung während der Arbeit.



Glossar

Authentifizierung Verifizierung der Identität eines Benutzers gegenüber einem System (vgl. [LS17]).

Autorisierung Eine Autorisierung berechtigt eine Person, IT-Komponente oder Anwendung zur Durchführung einer bestimmten Aktion (vgl. [BSIc]).

Brute-Force-Attacke Automatisierter Angriff, um Passwörter durch Ausprobieren herauszufinden..

Conversation Flow Tool Enthält JSON-Datei mit Dialogen, für die Szenarien eines VUI-Tests.

Conversation Prototyping Tool CPT ist ein auf Max basierte Anwendung, welche es erlaubt Prototypen von Conversational Interfaces (Voice-Only, Voice-First, Chatbot) zu erstellen und die User Experience nach dem Wizard of Oz-Verfahren zu testen.

Conversational User Interface Ein Conversational User Interface gilt als eine Benutzerschnittstelle, die den Ansatz der Konversationsführung zwischen Nutzer und System bildet. Die Kommunikation kann hierbei durch textueller oder auditiver Eingabe erfolgen.

Graphical User Interface Eine grafische Benutzeroberfläche, die zur Darstellung von visuellen Anzeigen zur Interaktion mit elektronischen Geräten dient (vgl. [Wha]).

Hot-Spots Markante Bereiche einer grafischen Darstellung.

Mock-Up Mock-Ups sind visuelle Simulationen oder Entwürfe (vgl. [Pea16]).



Glossar

Multimodal Conversational Interface Dieses setzt sich aus zwei den verschiedenen Begriffen 'multimodal' und 'conversational User Interface' zusammen und bietet verschiedene Modalitäten zur Konversationsführung zwischen Nutzer und System.

Native User Interface Benutzerschnittstelle um mit einem System über natürliche Interaktion zu kommunizieren.

Phrase Render Tool Das PRT liest JSON-Dateien ein und rendert in der Datei enthalten Voice-Responses von sprachbasierten Systemen in Audiofiles - und kann dabei Namen von Probanden mit angegebener Wahrscheinlichkeit einsetzen.

Picture-Superiority-Effect Der Effekt bestätigt eine bessere Einprägsamkeit einer grafischen Darstellung im Vergleich zu alphanumerischen Elementen (vgl [NRW76]).

Shoulder-Surfing Methode zur Spionage eines Benutzers bei der Betätigung eines elektronischen Geräts um die PIN oder Passwort zu erhalten (vgl. [Sho]).

Usability-Test Beim Usability-Test interagiert der repräsentative Benutzer in der Regel mit einem interaktiven System um verschiedene Aufgaben zu lösen. Hierbei sollen Usability- Probleme des Systems analysiert und identifiziert um spätere Implementierungsfehler zu vermeiden. Es wird gezielt auf das Verständnis der Erfordernisse an ein System examiniert und auf das Verhalten des Benutzers bei der Interaktion fokussiert (vgl. [Hop18]).

User Interface Benutzerschnittstelle um mit einem System zu interagieren..

Voice User Interface Benutzerschnittstelle um mit einem System über Sprache zu interagieren.

Voice-first Kommunikation mit einem System zunächst über Sprache, jedoch in Ausnahmefällen auch über manuellen Input möglich.

Voice-only Kommunikation mit einem System ausschließlich über Sprache.

WIMP Interaktionskonzept als technischer Layer in der Anwendung. 'Windows, Icons, Menü und Pointers' beschreibt hierbei die Modalität der Mensch-Maschinen-Schnittstelle.



Glossar

Wizard-Of-Oz Methode zum Test der Gebrauchstauglichkeit eines vorgestellten Systems mit natürlicher Sprache (vgl. [Wiz]).



Literaturverzeichnis

- [ado] adorsys GmbH & Co. KG. Company 2020.
- [BBGV15] Heiner Bubb, Klaus Bengler, Rainer E. Grünen, and Mark Vollrath. *Automobilergonomie*. ATZ/MTZ-Fachbuch. Springer Vieweg, 2015.
- [Ber18] André Berton. Mbux voice assistant: Versteht dich von selbst. <https://blog.daimler.com/2018/12/28/mbux-voice-assistant-hey-mercedes/>, December 2018. [Online; Abgerufen 03.11.2019].
- [Blo96] Greg E. Blonder. Graphical password, September 1996.
- [Bor17] Craig Borowski. Want a Memorable Toll-Free Number? Ask a Neuroscientist. <https://www.softwareadvice.com/resources/want-memorable-toll-free-number-ask-neuroscientist/>, 2017. [Online; Abgerufen 24.03.2020].
- [Bri] Briefmarken Jahrgang Deutschland 1953 postfrisch oder gestempelt - borek.de. <https://www.borek.de/briefmarken-deutschland-jahrgang-1953-komplett-postfrisch-gestempelt>. [Online; Abgerufen 08.01.2020].
- [BSIa] BSI-Biometrie Allgemeine Einführung-Grundsätzliche Funktionsweise Biometrischer Verfahren. <https://www.bsi.bund.de/DE/Themen/DigitaleGesellschaft/Biometrie/AllgemeineEinfuehrung/einfuehrung.html>. [Online; Abgerufen 03.04.2020].
- [BSIb] BSI-Für Bürger-Passwörter. https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Passwoerter/passwoerter/node_.html. [Online; Abgerufen 22.04.2020].
- [BSIc] BSI-IT-Grundschutz-Kompendium- Glossar. https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzKompendium/vorkapitel/Glossar_.html. [Online; Abgerufen 17.05.2020].
- [CBW04] Dean.P Chiang, Aaron M. Brooks, and David.H Weir. On the highway measures of driver glance behavior with an example automobile navigation system. pages 215–223, 2004.



Literaturverzeichnis

- [Cen10] Imperva Application D Center. Consumer password worst practices. Technical report, 2010.
- [CFBv09] Sonia Chiasson, Alain Forget, Robert Biddle, and P. C. van Oorschot. User interface design affects security: Patterns in click-based graphical passwords. *International Journal of Information Security*, pages 387–398, 2009.
- [Coa18] Dustin Coates. Talking to Computers | The Three Levels of Voice Interaction: Voice-First, Voice-Only, and Voice-Added. <https://www.talkingtocomputers.com/three-levels-voice-interaction>, 2018. [Online; Abgerufen 12.04.2020].
- [Cog] Cognition | Meaning of Cognition by Lexico. <https://www.lexico.com/definition/cognition>. [Online; Abgerufen 25.05.2020].
- [Com08] Commission Recommendation of 26 May 2008 on safe and efficient in-vehicle information and communication systems: Update of the European Statement of Principles on human-machine interface (notified under document number C(2008) 1742). Technical report, 2008.
- [CRZ11] Christine Chaloupka, Ralf Risser, and Wolf-Dietrich Zuzan. *Verkehrspychologie: Grundlagen und Anwendungen*. Facultas.WUV, 2011.
- [Cyc] Cycling '74. <https://cycling74.com/>. [Online; Abgerufen 30.05.2020].
- [Das18] Ritwik Dasgupta. *Voice User Interface Design: Moving from GUI to Mixed Modal Interaction*. Apress, 2018.
- [Del15] Delpixart. Städte Der Wortcollage Square Bei Nacht Stockfoto und mehr Bilder von Montage - Composite-Technik - iStock. <https://www.istockphoto.com/de/foto/st%C3%A4de-der-wort-collage-square-bei-nacht-gm486441966-72533367>, 2015. [Online; Abgerufen 08.01.2020].
- [DGV] DGVP Verkehrspychologie. <https://www.dgvp-verkehrspychologie.de/>. [Online; Abgerufen 29.03.2020].
- [DP00] Rachna Dhamija and Adrian Perrig. Déjà Vu: A user study using images for authentication. In *Proceedings of the 9th Conference on USENIX Security Symposium - Volume 9*. USENIX Association, 2000.
- [Esp18] Emily Esposito. Low-fidelity vs. high-fidelity prototyping. <https://www.invisionapp.com/inside-design/low-fi-vs-hi-fi-prototyping/>, 2018. [Online; Abgerufen 01.05.2020].



Literaturverzeichnis

- [Fod18] Federico Foderaro. Space Invaders remake in Max | Federico Foderaro - Amazing Max Stuff on Patreon. <https://www.patreon.com/posts/space-invaders-16951388>, 2018. [Online; Abgerufen 10.12.2020].
- [For14] Christian Forst. Sichere Authentifizierung – Teil I: Klassische Methoden. <https://conplore.com/sichere-authentifizierung-teil-i-klassische-methoden/>, 2014. [Online; Abgerufen 25.04.2020].
- [Fuj] Fujitsu Deutschland. <https://www.fujitsu.com/de/>. [Online; Abgerufen 25.05.2020].
- [Ger16] Dave Gershgorn. Carnegie Mellon made a special pair of glasses that lets you steal a digital identity — Quartz. <https://qz.com/823820/carnegie-mellon-made-a-special-pair-of-glasses-that-lets-you-steal-a-digital-identity/>, 2016.
- [Gra] Graphical User Interface | Definition of Graphical User Interface by Lexico. https://www.lexico.com/en/definition/graphical_user_interface. [Online; Abgerufen 29.05.2020].
- [GW16] Mrs. Aakansha S. Gokhale and Vijaya S. Waghmare. The Shoulder Surfing Resistant Graphical Password Authentication Technique. *Procedia Computer Science*, pages 490–498, 2016.
- [Hän18] Gerhard Hänel. Mautsysteme in Europa | IMPARGO. <https://impargo.de/blog/mautsysteme-in-europa>, 2018. [Online; Abgerufen 25.04.2020].
- [Haw18] Andrew J. Hawkins. Chevy owners can now pay for gas from inside their cars. <https://www.theverge.com/2018/4/18/17248282/chevrolet-shell-in-car-payment-gas>, 2018. [Online; Abgerufen 02.03.2020].
- [hel14] How to Use Picture Password in BlackBerry 10 OS version 10.2.1. <https://helpblog.blackberry.com/en/2014/02/how-to-use-picture-password-in-blackberry-10-os-version-10-2-1>, 2014. [Online; Abgerufen 10.04.2020].
- [Hin16] Michael Hines. Tech Before Its Time: 30 Years Ago Buick Put The First Touchscreen In A Car. <https://carbuzz.com/news/tech-before-its-time-30-years-ago-buick-put-the-first-touchscreen-in-a-car>, 2016. [Online; Abgerufen 14.04.2020].
- [HK17] Ralph Herkenhöner and Karsten Keusch. Biometrische Authentifizierung: Effektive Zutrittskontrolle. <https://www.lanline.de/it-security/effektive-zutrittskontrolle.213670.html>, 2017. [Online; Abgerufen 25.05.2020].



Literaturverzeichnis

- [HNE00] Joanne Harbluk, Ian Noy, and Moshe Eizenman. THE IMPACT OF INTERNAL DISTRACTION ON DRIVER VISUAL BEHAVIOR. 2000.
- [Hop18] Hans Georg Hopf. Usability - Test- Darstellung nach dem UXQB-CPUX-UT Curriculum. 2018.
- [How14] How Smart is BlackBerry Picture Password? Smart!! | Idealistically Caspan. <http://caspan.com/2014/02/how-smart-is-blackberry-picture-password-smart/>, 2014. [Online; Abgerufen 10.04.2020].
- [How17] How to use Picture Password on the BlackBerry KEYone | CrackBerry. <https://crackberry.com/how-use-picture-password-blackberry-keyone>, 2017. [Online; Abgerufen 03.11.2019].
- [HS88] Sandra G. Hart and Lowell E. Staveland. Development of NASA-TLX (Task Load Index): Results of Empirical and Theoretical Research. In Peter A. Hancock and Najmedin Meshkati, editors, *Advances in Psychology*, volume 52 of *Human Mental Workload*, pages 139–183. North-Holland, 1988.
- [InC20] In-Car Voice Assistant Consumer Adoption Report 2020 - Voicebot.ai. <https://voicebot.ai/in-car-voice-assistant-consumer-adoption-report-2020/>, 2020. [Online; Abgerufen 14.04.2020].
- [Int] Internetauftritt des Bundesbeauftragten für den Datenschutz und die Informationsfreiheit - Technische Anwendungen - Biometrie und Datenschutz. https://www.bfdi.bund.de/DE/Datenschutz/Themen/Technische_Anwendungen/TechnischeAnwendungenArtikel/BiometrieUndDatenschutz.html. [Online; Abgerufen 25.04.2020].
- [Jit] Jit.world Object Reference - Max 7 Documentation. <https://docs.cycling74.com/max7/refpages/jit.world>. [Online; Abgerufen 06.05.2020].
- [JMM⁺99] Ian Jermyn, Alain Mayer, Fabian Monrose, Michael K Reiter, and Aviel D Rubin. THE DESIGN AND ANALYSIS OF GRAPHICAL PASSWORDS. page 15, 1999.
- [Jsu] Jsui Object Reference - Max 7 Documentation. <https://docs.cycling74.com/max7/refpages/jsui>. [Online; Abgerufen 15.05.2020].
- [KAX11] Wazir Zada Khan, Mohammed Y. Aalsalem, and Yang Xiang. A Graphical Password Based System for Small Mobile Devices. *ArXiv*, 2011.



Literaturverzeichnis

- [KB81] Gideon Keren and Stan Baggen. Recognition models of alphanumeric characters. *Perception & Psychophysics*, pages 234–246, 1981.
- [Koh05] Chris Kohler. *Power-Up: How Japanese Video Games Gave the World an Extra Life*. BradyGames, 2005.
- [Kor19] Sven Korschinowski. In-Car-Payments als neuer Trend beim bargeldlosen Bezahlten. <https://www.der-bank-blog.de/in-car-payments/mobile-payment/37655183/>, 2019. [Online; Abgerufen 06.04.2020].
- [KV11] Josef F. Krems and Mark Vollrath. *Verkehrspsychologie: Ein Lehrbuch für Psychologen, Ingenieure und Informatiker*. Kohlhammer Verlag, 2011.
- [LS17] Stefan Luber and Peter Schmitz. Was ist Authentifizierung? <https://www.security-insider.de/was-ist-authentifizierung-a-617991/>, 2017.
- [Lyr] Lyrebird: Ultra-realistic voice cloning and text to speech | Descript. <https://www.descript.com/lyrebird-ai?source=lyrebird>. [Online; Abgerufen 04.04.2020].
- [Mas20] Mastercard Challenge | Voice Payments in Connected Cars. <https://www.madridinmotion.online/en/retos/mastercard-en-2/>, 2020. [Online; Abgerufen 14.04.2020].
- [MDH⁺07] Michael A. Mollenhauer, Thomas A. Dingus, Jonathan M. Hankey, Cher Carney, and Vicki L. Neale. Index - Development Of Human Factors Guidelines For Advanced Traveler Information Systems (ATIS) And Commercial Vehicle Operations (CVO): Display Formats And Commercial Vehicle Operator (CVO) Workload, September 1997 - FHWA-RD-96-152. <https://www.fhwa.dot.gov/publications/research/safety/96152/index.cfm>, 2007. [Online; Abgerufen 25.05.2020].
- [Met09] Barbara Metz. *Worauf Achtet Der Fahrer? Steuerung Der Aufmerksamkeit Beim Fahren Mit Visuellen Nebenaufgaben*. Inaugural-Dissertation, Julius-Maximilians-Universität Würzburg, 2009.
- [Mil07] Brian Milligan. The man who invented the cash machine. 2007.
- [Mis14] Misidentification of Alphanumeric Symbols. <https://www.ismp.org/resources/misidentification-alphanumeric-symbols>, 2014. [Online; Abgerufen 04.05.2020].



Literaturverzeichnis

- [New09] New york times square-terabass.jpg – Wikipedia. https://commons.wikimedia.org/wiki/File>New_york_times_square-terabass.jpg, 2009. [Online; Abgerufen 18.12.2019].
- [NRW76] Douglas L. Nelson, Valerie S. Reed, and John R. Walling. Pictorial superiority effect. *Journal of Experimental Psychology: Human Learning and Memory*, pages 523–528, 1976.
- [NT18] Nicole Nitsche and Kilian Thalhammer. Payment im Auto - die neue Kundenschnittstelle der Zukunft. <https://paymentandbanking.com/payment-im-auto/>, 2018. [Online; Abgerufen 12.02.2020].
- [Obe13] Ankit Oberoi. How Typography Affects Readers. <https://www.adpushup.com/blog/how-typography-affects-readers/>, 2013. [Online; Abgerufen 04.05.2020].
- [Off18] ID Guard Offline. How to Enable Sign-in with iOS Biometric Authentication. <https://medium.com/@IDGuardOffline/how-to-enable-sign-in-with-ios-biometric-authentication-5c7249ba5b00>, 2018. [Online; Abgerufen 04.05.2020].
- [OPBS15] Alexander Osterwalder, Yves Pigneur, Greg Bernarda, and Alan Smith. *Value Proposition Design: Entwickeln Sie Produkte und Services, die Ihre Kunden wirklich wollen. Die Fortsetzung des Bestsellers Business Model Generation!* Campus Verlag, 2015.
- [Pas] Password | Definition of Password by Lexico. <https://www.lexico.com/en/definition/password>. [Online; Abgerufen 18.05.2020].
- [Pas20] Passwort. <https://de.wikipedia.org/w/index.php?title=Passwort&oldid=200496389>, 2020. [Online; Abgerufen 14.04.2020].
- [Pea16] Cathy Pearl. *Designing Voice User Interfaces*. 2016.
- [Per20] Sarah Perez. ‘Alexa, pay for gas’ command to work at over 11,500 Exxon and Mobil stations this year | TechCrunch. <https://techcrunch.com/2020/01/06/alexapay-for-gas-command-to-work-at-over-11500-exxon-and-mobil-stations-this-year/>, 2020. [Online; Abgerufen 14.04.2020].
- [Pet18] Russell Peters. Chevrolet Test Program Allows Drivers to Pay for Gas Using Infotainment Systems | Carfilia - Auto Blog and Auto Reviews. <http://www.carfilia.com/auto-news/chevrolet-test-program-allows-drivers-to-pay-for-gas-using-infotainment-systems/>, 2018. [Online; Abgerufen 22.04.2020].



Literaturverzeichnis

- [PIN] PIN Defintion. https://www.oxfordlearnersdictionaries.com/definition/english/pin_3. [Online; Abgerufen 18.05.2020].
- [PIN17] PIN vs. Password: What's the Difference? <https://www.keepersecurity.com/blog/2017/03/07/pin-vs-password-whats-the-difference/>, 2017. [Online; Abgerufen 22.04.2020].
- [Por20] Jon Porter. Google tests voice matching to secure google assistant purchases. <https://www.theverge.com/2020/5/26/21270222/google-assistant-voice-match-purchases-in-app-google-play-shopping>, 2020. [Online; Abgerufen 04.06.2020].
- [RA17] Gedeon Rauch and Stephan Augsten. Was ist eine GUI? <https://www.dev-insider.de/was-ist-eine-gui-a-651868/>, 2017. [Online; Abgerufen 29.05.2020].
- [RDA09] Karen Renaud and Antonella De Angeli. Visual passwords: Cure-all or snake-oil? *Communications of the ACM*, pages 135–140, 2009.
- [RF16] Michael Richter and Markus D. Flückiger. *Usability und UX kompakt: Produkte für Menschen*. IT kompakt. Springer Vieweg, fourth edition, 2016.
- [Rou07] Margaret Rouse. What is graphical password or graphical user authentication (GUA)? - Definition from WhatIs.com. <https://searchsecurity.techtarget.com/definition/graphical-password>, 2007. [Online; Abgerufen 24.03.2020].
- [SB02] Leonardo Sobrado and Jean-Camille Birget. Graphical Passwords. <https://rutgersscholar.libraries.rutgers.edu/volume04/sobrbirg/sobrbirg.htm>, 2002. [Online; Abgerufen 14.04.2020].
- [SB13] Elizabeth Stobert and Robert Biddle. Memory retrieval and graphical passwords. In *Proceedings of the Ninth Symposium on Usable Privacy and Security - SOUPS '13*, Newcastle, United Kingdom, 2013. ACM Press.
- [SBBR16] Mahmood Sharif, Sruti Bhagavatula, Lujo Bauer, and Michael K. Reiter. Accessorize to a Crime: Real and Stealthy Attacks on State-of-the-Art Face Recognition. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security - CCS'16*, pages 1528–1540, Vienna, Austria, 2016. ACM Press.
- [Sho] Shoulder Surfing | Meaning of Shoulder Surfing by Lexico. https://www.lexico.com/definition/shoulder_surfing. [Online; Abgerufen 17.04.2020].



Literaturverzeichnis

- [Sur] Surface Pro 3-Features. <https://support.microsoft.com/de-de/help/4023445/surface-pro-3-features>. [Online; Abgerufen 06.05.2020].
- [TA08] Hai Tao and Carlisle Adams. Pass-Go: A Proposal to Improve the Usability of Graphical Passwords. 2008.
- [VoC16] #VoCo. Adobe Audio Manipulator Sneak Peak with Jordan Peele | Adobe Creative Cloud, 2016.
- [war] Zwei-faktor-authentifikation - so funktioniert sie - stiftung warentest. <https://www.test.de/Internetsicherheit-Yubikey-kleiner-Schlüssel-für-großen-Schutz-4807972-4807984/>. [Online; Abgerufen 17.04.2020].
- [Wha] What is a Graphical User Interface? Definition and FAQs | OmniSci. <https://www.omnisci.com/technical-glossary/graphical-user-interface>. [Online; Abgerufen 17.05.2020].
- [Wiz] Wizard of Oz | Usability Body of Knowledge. <https://www.usabilitybok.org/wizard-of-oz>. [Online; Abgerufen 07.05.2020].
- [WW11] Daniel Wigdor and Dennis Wixon. *Brave NUI World: Designing Natural User Interfaces for Touch and Gesture*. Morgan Kaufmann, Burlington, Mass, 2011.
- [ZAD00] Helmut T. Zwahlen, Charles C. Adams, and David P. DeBals. Safety Aspects of CRT Touch Panel Controls in Automobiles. In *Vision in Vehicles II. Second International Conference on Vision in Vehicles*. Applied Vision AssociationErgonomics SocietyAssociation of Optometrists, 1988/00/00.



A Anhang

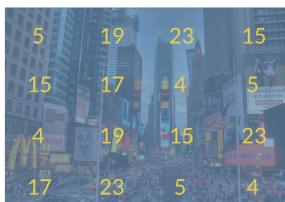
A Anhang

A.1 Vollständiges Storyboard der Konversation

Szenario 1: Der User möchte mit dem Auto nach Österreich fahren. Aus diesem Grund möchte er für 7 Tage ein Mautticket kaufen.		
User	Car Assistant	API Screen
Hallo Auto ich brauche bitte ein Mautticket!		
	Ok, Ich sehe, du bist gerade Nähe der österreichischen Grenze. Möchtest Du ein Pickerl für Österreich haben?	
Ja, bitte.	Es gibt Tickets für eine Zeitspanne von 10 Tagen, zwei Monaten und einem Jahr. Welches möchtest du erwerben?	



A Anhang

	Ich möchte bitte die Vignette für 10 Tage kaufen.	 <p>10 Tage 9,40 Euro Österreich</p>
		 <p>5 19 23 15 15 17 4 5 4 19 15 23 17 23 5 4</p>
Ich hätte jetzt das Passwort.	Bitte teile mir den Passcode nun mit.	 <p>Österreich</p>
315	Vielen Dank, deine Zahlung wurde getätigert. Du darfst nun offiziell mit dem Kennzeichen LAU CD243 nach Österreich fahren. Dieses Ticket gilt von 24.11.19 12:30 Uhr bis 03.12.2019 12:30 Uhr. Viel Spaß und eine gute Fahrt.	 <p>Österreich</p>



A Anhang

A.2 Mock-Up-Screens des Prototypen



Startbildschirm



Ansicht Nähe österreichischer Grenze



Auswahl der Zeitspanne Mauttickets



Auswahl 10 Tage Mautticket



Auswahl zwei Monate Mautticket



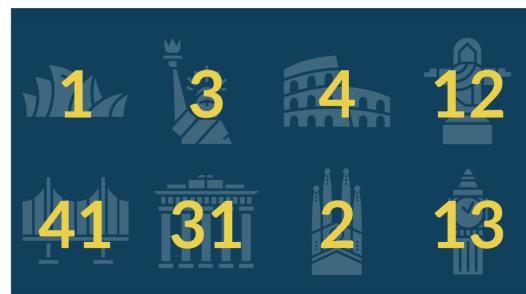
Auswahl ein Jahr Mautticket



A Anhang



Layer Passcode 1



Layer Passcode 2



Layer Passcode 3



Layer Passcode 4



Anzeige falscher Passcode



Anzeige zur Passcode



Bestätigung der Zahlung des Tickets



Zurück zum Startbildschirm



A.3 Aufgabenstellung und Szenario für den Usability-Test

Szenario

Beschreibung

Du fährst mit deinem Auto nach Österreich. Aus diesem Grund möchtest du dir ein Maut Ticket für 10 Tage kaufen. Das Ticket kaufst du während du fährst über den Sprachassistenten in deinem Fahrzeug.

Aufgabenstellung

Nutze den Sprachassistenten um ein Mautticket für 10 Tage zu kaufen.

Weitere Informationen

- Dein Auto hört auf „Hallo Auto“
- Das Picture Passcode läuft ohne zeitliche Beschränkung durch.
- Deine Secrets sind:



Big Ben in London



Moai auf den Osterinseln



A Anhang

A.4 Ergebnisse und Gruppenteilung der Umfrage für den Usability-Test

Probandenumfrage

	Termin	Fahrpraxis	Multitaskingfähigkeit	Ablenkbarkeit	Praxis mit Voice	Gruppe
1.1 (männlich)						B (gemütlich)
1.2 (männlich)						A (flott)
2.1 (weiblich)						A (flott)
2.2 (weiblich)						B (gemütlich)
3.1 (männlich)						A (flott)
3.2 (weiblich)						B (gemütlich)
4.1 (weiblich)						A (gemütlich)
4.2 (männlich)						B (flott)
5.1 (männlich)						A (gemütlich)
5.2 (weiblich)						B (flott)
6.1 (weiblich)						A (gemütlich)
6.2 (männlich)						B (flott)
	Routiniert	Gut	Kaum	Viel		
	Mittel	Mittel	Mittel	Mittel		
	Nicht routiniert	Schlecht	Ziemlich leicht	Kaum		

A ist Mittelkonsole

B ist Instrumentenfeld/Tacho/Lenkrad



A Anhang

A.5 Einverständniserklärung zur Audio-Aufnahme der Probanden

Usability-Test zur Erfassung von Erfahrungen und Erwartungen in Hinblick auf die Nutzung eines sprachgesteuerten Chatbots

Bei adorsys GmbH & Co. KG beschäftigen wir uns, im Rahmen einer Bachelorarbeit an der Technischen Hochschule Nürnberg, mit dem Design, der Bedienung und der Funktionalität eines Sprachassistenten, der im Automotive Bereich bei der Bezahlung während der Autofahrt unterstützend fungieren soll.

Hierbei geht es um:

- die Benutzerführung über Spracheingabe
- die Bedienbarkeit der Anwendung im Kontext Fahren

Hierfür führen wir einen Usability-Test mit einem von uns entwickelten Prototyping-Tool durch. Das Ziel ist es, mehr über Erwartungen von potentiellen Anwendern und die von ihnen am angenehmsten empfundene Herangehensweise bei der Nutzung des multimodalem Conversational Interfaces zu erfahren, und diese Erkenntnisse in die Entwicklung des Assistenten einfließen lassen zu können.

Alle Fragen und Aufgaben in diesem Usability-Test beziehen sich auf die Einstufung und Verbesserung des Interfaces und nicht darauf, die Fähigkeiten und Fertigkeiten des Probanden / der Probandin zu testen. Alle Angaben werden vertraulich behandelt und ausschließlich anonymisiert verarbeitet und archiviert. Weiterhin werden diese Angaben nur im Rahmen der Entwicklung dieses Chatbots und in hierauf bezogene Präsentationen verwendet.

Test-Setup

Method Thinking-aloud

Dauer ca. 45 Minuten

Durchführende

Unternehmen adorsys GmbH & Co. KG

Projektleitung Steffen Blümm, Technical Lead CUI

Datum: 16.03.2020

Einverständniserklärung

Ich erkläre mich dazu bereit, im Rahmen des oben beschriebenen Projekts an einem Usability-Test teilzunehmen. Ich wurde über die Ziele des Projekts informiert. Ich kann den Test jederzeit abbrechen, weitere Tests ablehnen und meine Einwilligung in eine Aufzeichnung und Niederschrift des Tests jederzeit zurückziehen, ohne dass mir dadurch irgendwelche Nachteile entstehen.

Ich bin damit einverstanden, dass der Test mit einem Aufnahmegerät aufgezeichnet und sodann von den MitarbeiterInnen der adorsys ausgewertet wird. Für die Auswertung des Usability-Tests werden alle Angaben zu meiner Person aus dem Text entfernt und/oder anonymisiert. Mir wurde außerdem versichert, dass der Test in Veröffentlichungen nur in Ausschnitten zitiert wird, um sicherzustellen, dass ich auch durch die Reihenfolge von im Test erwähnten Ereignissen nicht für Dritte erkennbar sein werde.

Datum, Unterschrift der Testperson

A.6 Ergebnisse des NASA-TLX

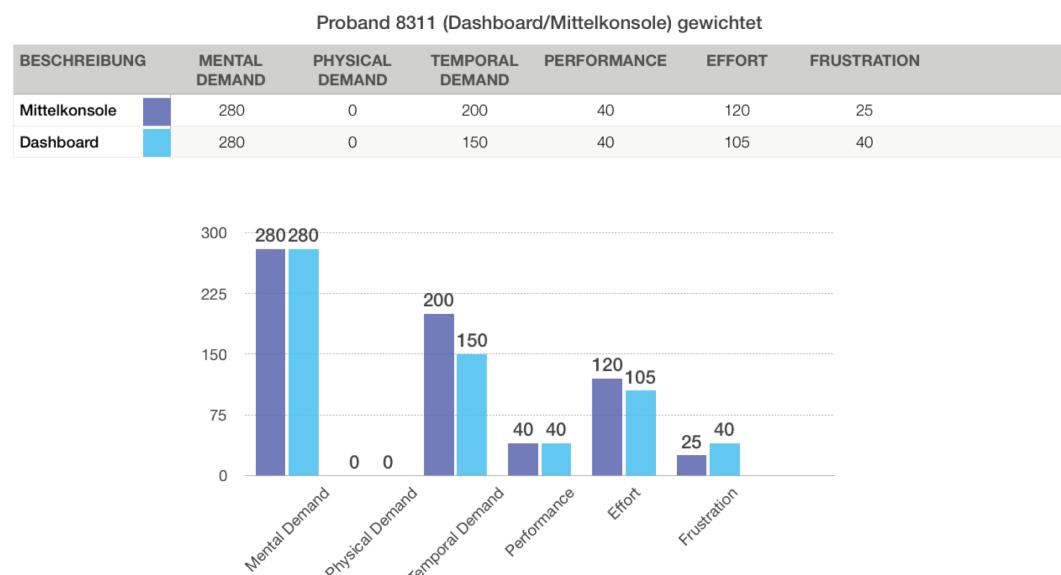
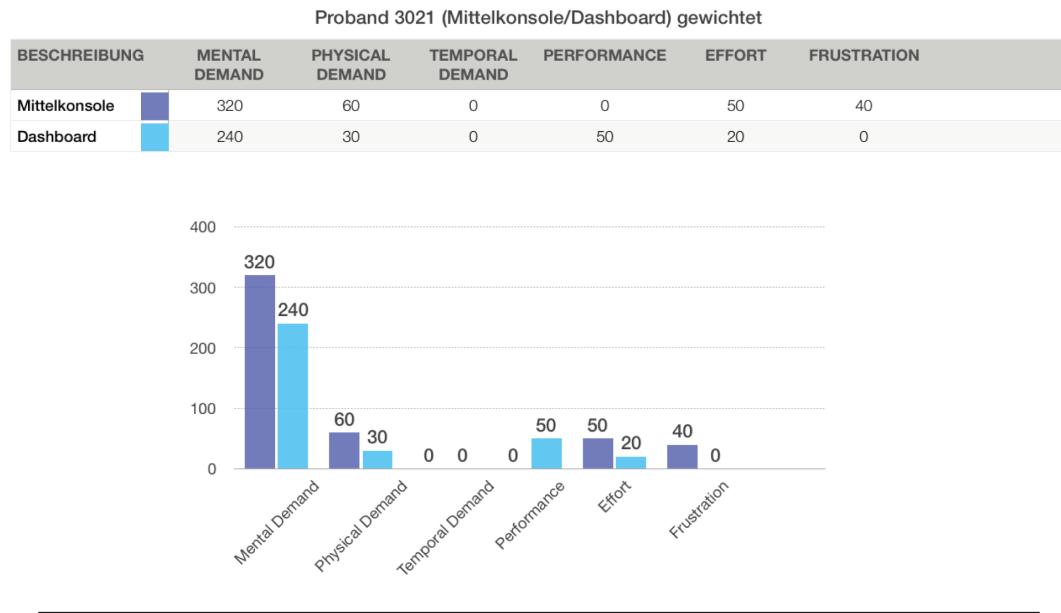
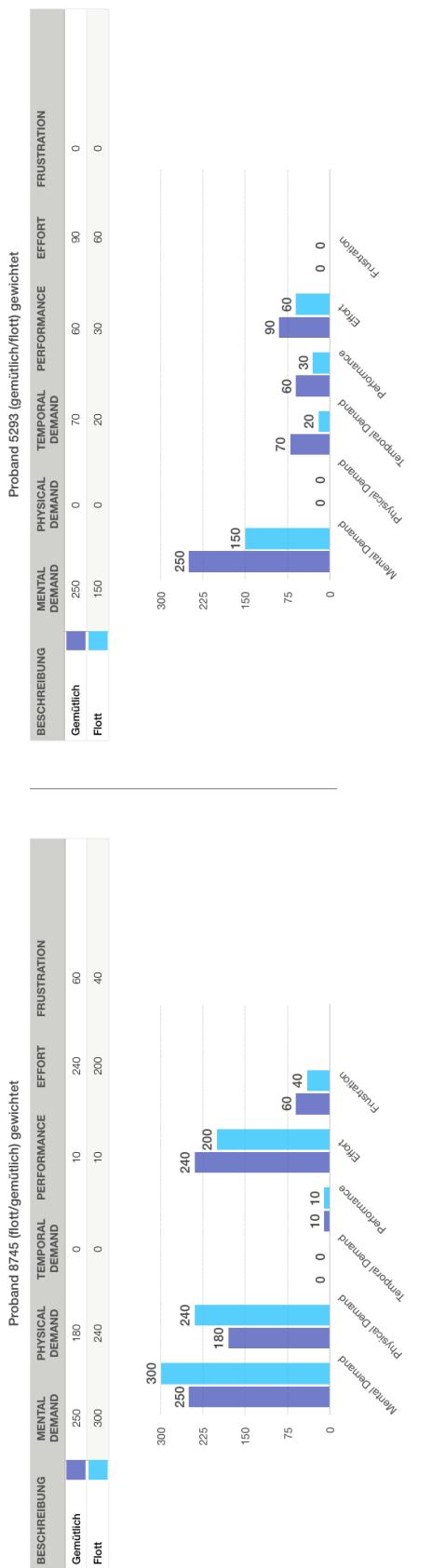


Abbildung A.1: Ergebnisse der Pre-Tests

B Test Dashboard



A Test Mittelkonsole

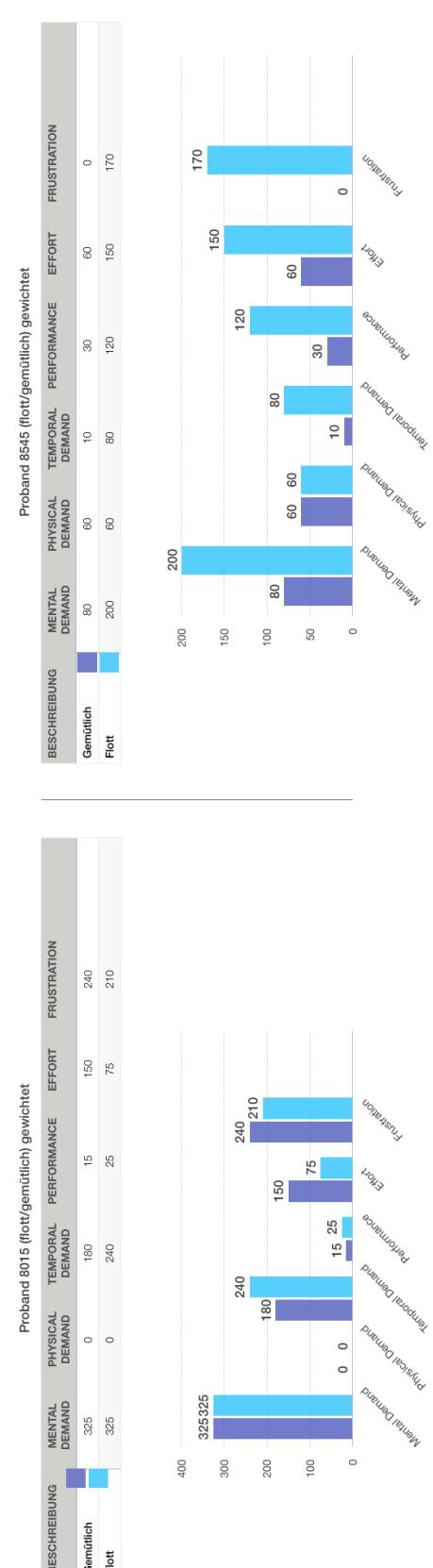
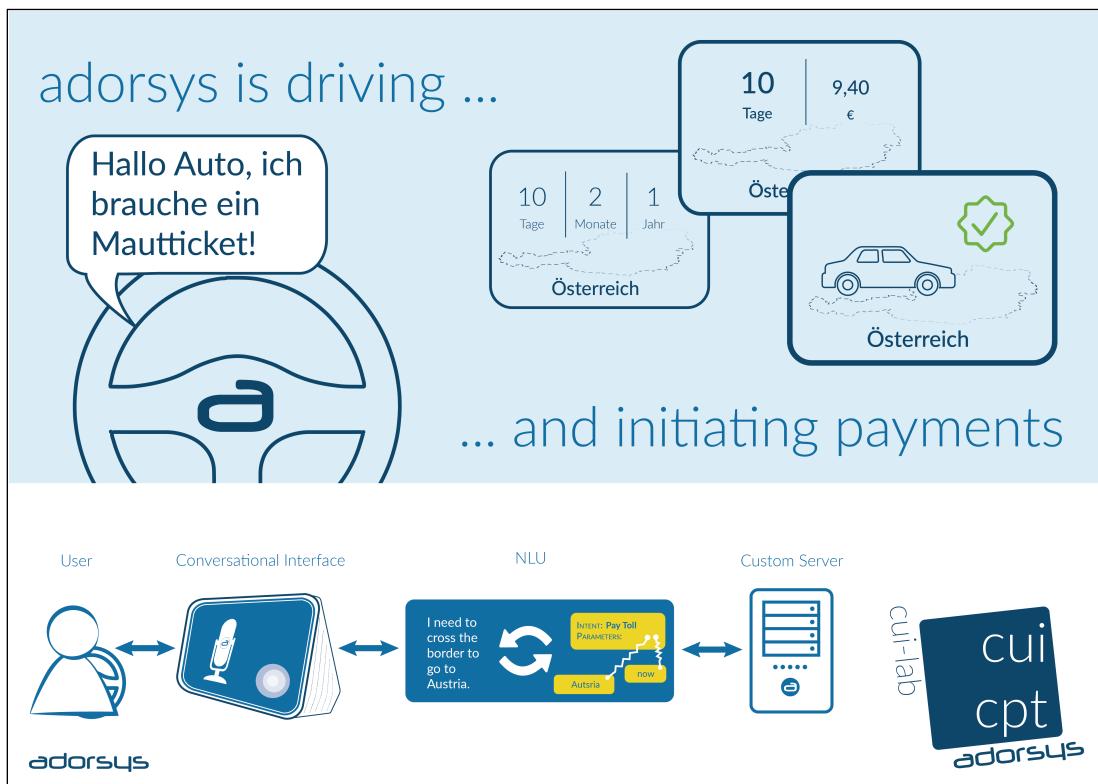


Abbildung A.2: Ergebnisse der offiziellen Usability-Tests



A Anhang

A.7 Plakat zur Raumgestaltung der Usability-Testessen





A Anhang

A.8 Fragebogen zum Usability-Test

Fragebogen:

Tester:

UsabilityTest: A / B

Denkst du der Passcode ist eine sichere Methode um Bezahlungen im Auto zu machen?

Welche Dinge würdest du kaufen oder würdest du benutzen?

Würdest du lieber Anfang der Fahrt das Passcode machen oder während dem Fahren?

Vergleiche die gestellte Aufgabe, welche eine ähnliche/leichtere/schwerere Beanspruchung während dem Fahren verlangt.

Leichtere Aufgabe:

Mittel:

Schwerere Aufgabe:

Empfindest du die im Test simulierten Konditionen mit der Wirklichkeit des Autofahrens vergleichbar?

Du bist nicht alleine im Auto, würdest du den Bezahl Methode verwenden?

Wie viel Wert legst du beim Autokauf, auf Features?

Es herrscht gerade eine schwierige Verkehrssituation (zb. Starker Regen, viel Verkehr). Du möchtest eine Zahlung durchführen. Das System schlägt dir aber vor, deine Bezahlung später zu machen, da gerade eine schwierige Verkehrssituation herrscht. Findest du das gut?



A.9 Quellcode der Passcode-Generierung

```
1 inlets = 1;
2 outlets = 3;
3
4 var firstIconNames = 10;
5 var problemNumber = ["0"];
6
7 function getRandomNumbers() {
8     number = Math.floor(Math.random() * (999 - 100 + 1) + 100);
9     digits = number.toString().split("");
10    for (var i = 0; i < digits.length - 1; i++) {
11        if ((digits[i] >= digits[i + 1]) /*get increasing Number*/ || (digits.
12            indexOf('0') > -1) /*get 0 Number*/ || (digits[0] == digits[i]) &&
13            digits[i] == digits[i + 1] /*get repNumber*/) {
14            post('wrongNumbers' + digits)
15        } else {
16            correctNumber = digits;
17        }
18    }
19
20    function containsZero(digits) {
21        for (var i = 0; i < digits.length; i++) {
22            if (digits[i] == '0') {
23                return true;
24            }
25        }
26
27    function containsRepeatedNumbers(digits) {
28        targetDigit = digits[0];
29        repeatedCount = 1;
30        for (var i = 1; i < digits.length; i++) {
31            if (digits[i] === targetDigit) {
32                post(digits[i], repeatedCount);
33                repeatedCount++
34            }
35        }
36        return repeatedCount == digits.length;
37    }
38
39    function adjacentDigits(digits) {
40        differenceArray = new Array();
41        for (var i = 0; i < digits.length - 1; i++) {
42            differenceArray.push(digits[i] - digits[i + 1])
43        }
44    }
45
46    function containsConsecutiveDuplicates(digits) {
47        for (var i = 0; i < digits.length - 1; i++) {
48            if (digits[i] == digits[i + 1]) {
49                return true;
50            }
51        }
52        return false;
53    }
54
55    function containsConsecutiveZeros(digits) {
56        for (var i = 0; i < digits.length - 1; i++) {
57            if (digits[i] == '0' && digits[i + 1] == '0') {
58                return true;
59            }
60        }
61        return false;
62    }
63
64    function containsConsecutiveRepeatingDigits(digits) {
65        for (var i = 0; i < digits.length - 1; i++) {
66            if (digits[i] == digits[i + 1]) {
67                return true;
68            }
69        }
70        return false;
71    }
72
73    function containsConsecutiveNonZeroDigits(digits) {
74        for (var i = 0; i < digits.length - 1; i++) {
75            if (digits[i] != '0' && digits[i + 1] != '0') {
76                return true;
77            }
78        }
79        return false;
80    }
81
82    function containsConsecutiveOddDigits(digits) {
83        for (var i = 0; i < digits.length - 1; i++) {
84            if (digits[i] % 2 != 0 && digits[i + 1] % 2 != 0) {
85                return true;
86            }
87        }
88        return false;
89    }
90
91    function containsConsecutiveEvenDigits(digits) {
92        for (var i = 0; i < digits.length - 1; i++) {
93            if (digits[i] % 2 == 0 && digits[i + 1] % 2 == 0) {
94                return true;
95            }
96        }
97        return false;
98    }
99
100   function containsConsecutivePrimes(digits) {
101        for (var i = 0; i < digits.length - 1; i++) {
102            if (isPrime(digits[i]) && isPrime(digits[i + 1])) {
103                return true;
104            }
105        }
106        return false;
107    }
108
109    function containsConsecutiveNonPrimes(digits) {
110        for (var i = 0; i < digits.length - 1; i++) {
111            if (!isPrime(digits[i]) && !isPrime(digits[i + 1])) {
112                return true;
113            }
114        }
115        return false;
116    }
117
118    function containsConsecutiveSquares(digits) {
119        for (var i = 0; i < digits.length - 1; i++) {
120            if (isSquare(digits[i]) && isSquare(digits[i + 1])) {
121                return true;
122            }
123        }
124        return false;
125    }
126
127    function containsConsecutiveNonSquares(digits) {
128        for (var i = 0; i < digits.length - 1; i++) {
129            if (!isSquare(digits[i]) && !isSquare(digits[i + 1])) {
130                return true;
131            }
132        }
133        return false;
134    }
135
136    function containsConsecutiveCubes(digits) {
137        for (var i = 0; i < digits.length - 1; i++) {
138            if (isCube(digits[i]) && isCube(digits[i + 1])) {
139                return true;
140            }
141        }
142        return false;
143    }
144
145    function containsConsecutiveNonCubes(digits) {
146        for (var i = 0; i < digits.length - 1; i++) {
147            if (!isCube(digits[i]) && !isCube(digits[i + 1])) {
148                return true;
149            }
150        }
151        return false;
152    }
153
154    function containsConsecutiveFibonacciNumbers(digits) {
155        for (var i = 0; i < digits.length - 1; i++) {
156            if (isFibonacci(digits[i]) && isFibonacci(digits[i + 1])) {
157                return true;
158            }
159        }
160        return false;
161    }
162
163    function containsConsecutiveNonFibonacciNumbers(digits) {
164        for (var i = 0; i < digits.length - 1; i++) {
165            if (!isFibonacci(digits[i]) && !isFibonacci(digits[i + 1])) {
166                return true;
167            }
168        }
169        return false;
170    }
171
172    function containsConsecutivePrimeFactors(digits) {
173        for (var i = 0; i < digits.length - 1; i++) {
174            if (isPrimeFactor(digits[i], digits[i + 1])) {
175                return true;
176            }
177        }
178        return false;
179    }
180
181    function containsConsecutiveNonPrimeFactors(digits) {
182        for (var i = 0; i < digits.length - 1; i++) {
183            if (!isPrimeFactor(digits[i], digits[i + 1])) {
184                return true;
185            }
186        }
187        return false;
188    }
189
190    function containsConsecutiveCompositeFactors(digits) {
191        for (var i = 0; i < digits.length - 1; i++) {
192            if (isCompositeFactor(digits[i], digits[i + 1])) {
193                return true;
194            }
195        }
196        return false;
197    }
198
199    function containsConsecutiveNonCompositeFactors(digits) {
200        for (var i = 0; i < digits.length - 1; i++) {
201            if (!isCompositeFactor(digits[i], digits[i + 1])) {
202                return true;
203            }
204        }
205        return false;
206    }
207
208    function containsConsecutiveTwinPrimes(digits) {
209        for (var i = 0; i < digits.length - 1; i++) {
210            if (isTwinPrime(digits[i], digits[i + 1])) {
211                return true;
212            }
213        }
214        return false;
215    }
216
217    function containsConsecutiveNonTwinPrimes(digits) {
218        for (var i = 0; i < digits.length - 1; i++) {
219            if (!isTwinPrime(digits[i], digits[i + 1])) {
220                return true;
221            }
222        }
223        return false;
224    }
225
226    function containsConsecutiveTwinSquares(digits) {
227        for (var i = 0; i < digits.length - 1; i++) {
228            if (isTwinSquare(digits[i], digits[i + 1])) {
229                return true;
230            }
231        }
232        return false;
233    }
234
235    function containsConsecutiveNonTwinSquares(digits) {
236        for (var i = 0; i < digits.length - 1; i++) {
237            if (!isTwinSquare(digits[i], digits[i + 1])) {
238                return true;
239            }
240        }
241        return false;
242    }
243
244    function containsConsecutiveTwinCubes(digits) {
245        for (var i = 0; i < digits.length - 1; i++) {
246            if (isTwinCube(digits[i], digits[i + 1])) {
247                return true;
248            }
249        }
250        return false;
251    }
252
253    function containsConsecutiveNonTwinCubes(digits) {
254        for (var i = 0; i < digits.length - 1; i++) {
255            if (!isTwinCube(digits[i], digits[i + 1])) {
256                return true;
257            }
258        }
259        return false;
260    }
261
262    function containsConsecutiveTwinFibonacciNumbers(digits) {
263        for (var i = 0; i < digits.length - 1; i++) {
264            if (isTwinFibonacci(digits[i], digits[i + 1])) {
265                return true;
266            }
267        }
268        return false;
269    }
270
271    function containsConsecutiveNonTwinFibonacciNumbers(digits) {
272        for (var i = 0; i < digits.length - 1; i++) {
273            if (!isTwinFibonacci(digits[i], digits[i + 1])) {
274                return true;
275            }
276        }
277        return false;
278    }
279
280    function containsConsecutiveTwinPrimesAndTwinSquares(digits) {
281        for (var i = 0; i < digits.length - 1; i++) {
282            if (isTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
283                return true;
284            }
285        }
286        return false;
287    }
288
289    function containsConsecutiveNonTwinPrimesAndTwinSquares(digits) {
290        for (var i = 0; i < digits.length - 1; i++) {
291            if (!isTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
292                return true;
293            }
294        }
295        return false;
296    }
297
298    function containsConsecutiveTwinPrimesAndTwinCubes(digits) {
299        for (var i = 0; i < digits.length - 1; i++) {
300            if (isTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
301                return true;
302            }
303        }
304        return false;
305    }
306
307    function containsConsecutiveNonTwinPrimesAndTwinCubes(digits) {
308        for (var i = 0; i < digits.length - 1; i++) {
309            if (!isTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
310                return true;
311            }
312        }
313        return false;
314    }
315
316    function containsConsecutiveTwinPrimesAndTwinFibonacciNumbers(digits) {
317        for (var i = 0; i < digits.length - 1; i++) {
318            if (isTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
319                return true;
320            }
321        }
322        return false;
323    }
324
325    function containsConsecutiveNonTwinPrimesAndTwinFibonacciNumbers(digits) {
326        for (var i = 0; i < digits.length - 1; i++) {
327            if (!isTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
328                return true;
329            }
330        }
331        return false;
332    }
333
334    function containsConsecutiveTwinPrimesAndTwinPrimes(digits) {
335        for (var i = 0; i < digits.length - 1; i++) {
336            if (isTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
337                return true;
338            }
339        }
340        return false;
341    }
342
343    function containsConsecutiveNonTwinPrimesAndTwinPrimes(digits) {
344        for (var i = 0; i < digits.length - 1; i++) {
345            if (!isTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
346                return true;
347            }
348        }
349        return false;
350    }
351
352    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
353        for (var i = 0; i < digits.length - 1; i++) {
354            if (isTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
355                return true;
356            }
357        }
358        return false;
359    }
360
361    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
362        for (var i = 0; i < digits.length - 1; i++) {
363            if (!isTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
364                return true;
365            }
366        }
367        return false;
368    }
369
370    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
371        for (var i = 0; i < digits.length - 1; i++) {
372            if (isTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
373                return true;
374            }
375        }
376        return false;
377    }
378
379    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
380        for (var i = 0; i < digits.length - 1; i++) {
381            if (!isTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
382                return true;
383            }
384        }
385        return false;
386    }
387
388    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
389        for (var i = 0; i < digits.length - 1; i++) {
390            if (isTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
391                return true;
392            }
393        }
394        return false;
395    }
396
397    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
398        for (var i = 0; i < digits.length - 1; i++) {
399            if (!isTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
400                return true;
401            }
402        }
403        return false;
404    }
405
406    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
407        for (var i = 0; i < digits.length - 1; i++) {
408            if (isTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
409                return true;
410            }
411        }
412        return false;
413    }
414
415    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
416        for (var i = 0; i < digits.length - 1; i++) {
417            if (!isTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
418                return true;
419            }
420        }
421        return false;
422    }
423
424    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
425        for (var i = 0; i < digits.length - 1; i++) {
426            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
427                return true;
428            }
429        }
430        return false;
431    }
432
433    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
434        for (var i = 0; i < digits.length - 1; i++) {
435            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
436                return true;
437            }
438        }
439        return false;
440    }
441
442    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
443        for (var i = 0; i < digits.length - 1; i++) {
444            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
445                return true;
446            }
447        }
448        return false;
449    }
450
451    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
452        for (var i = 0; i < digits.length - 1; i++) {
453            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
454                return true;
455            }
456        }
457        return false;
458    }
459
460    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
461        for (var i = 0; i < digits.length - 1; i++) {
462            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
463                return true;
464            }
465        }
466        return false;
467    }
468
469    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
470        for (var i = 0; i < digits.length - 1; i++) {
471            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
472                return true;
473            }
474        }
475        return false;
476    }
477
478    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
479        for (var i = 0; i < digits.length - 1; i++) {
480            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
481                return true;
482            }
483        }
484        return false;
485    }
486
487    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
488        for (var i = 0; i < digits.length - 1; i++) {
489            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
490                return true;
491            }
492        }
493        return false;
494    }
495
496    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
497        for (var i = 0; i < digits.length - 1; i++) {
498            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
499                return true;
500            }
501        }
502        return false;
503    }
504
505    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
506        for (var i = 0; i < digits.length - 1; i++) {
507            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
508                return true;
509            }
510        }
511        return false;
512    }
513
514    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
515        for (var i = 0; i < digits.length - 1; i++) {
516            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
517                return true;
518            }
519        }
520        return false;
521    }
522
523    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
524        for (var i = 0; i < digits.length - 1; i++) {
525            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
526                return true;
527            }
528        }
529        return false;
530    }
531
532    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
533        for (var i = 0; i < digits.length - 1; i++) {
534            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
535                return true;
536            }
537        }
538        return false;
539    }
540
541    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
542        for (var i = 0; i < digits.length - 1; i++) {
543            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
544                return true;
545            }
546        }
547        return false;
548    }
549
550    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
551        for (var i = 0; i < digits.length - 1; i++) {
552            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
553                return true;
554            }
555        }
556        return false;
557    }
558
559    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
560        for (var i = 0; i < digits.length - 1; i++) {
561            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
562                return true;
563            }
564        }
565        return false;
566    }
567
568    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
569        for (var i = 0; i < digits.length - 1; i++) {
570            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
571                return true;
572            }
573        }
574        return false;
575    }
576
577    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
578        for (var i = 0; i < digits.length - 1; i++) {
579            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
580                return true;
581            }
582        }
583        return false;
584    }
585
586    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
587        for (var i = 0; i < digits.length - 1; i++) {
588            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
589                return true;
590            }
591        }
592        return false;
593    }
594
595    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
596        for (var i = 0; i < digits.length - 1; i++) {
597            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
598                return true;
599            }
600        }
601        return false;
602    }
603
604    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
605        for (var i = 0; i < digits.length - 1; i++) {
606            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
607                return true;
608            }
609        }
610        return false;
611    }
612
613    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
614        for (var i = 0; i < digits.length - 1; i++) {
615            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
616                return true;
617            }
618        }
619        return false;
620    }
621
622    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
623        for (var i = 0; i < digits.length - 1; i++) {
624            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
625                return true;
626            }
627        }
628        return false;
629    }
630
631    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
632        for (var i = 0; i < digits.length - 1; i++) {
633            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
634                return true;
635            }
636        }
637        return false;
638    }
639
640    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
641        for (var i = 0; i < digits.length - 1; i++) {
642            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
643                return true;
644            }
645        }
646        return false;
647    }
648
649    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
650        for (var i = 0; i < digits.length - 1; i++) {
651            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
652                return true;
653            }
654        }
655        return false;
656    }
657
658    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
659        for (var i = 0; i < digits.length - 1; i++) {
660            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
661                return true;
662            }
663        }
664        return false;
665    }
666
667    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
668        for (var i = 0; i < digits.length - 1; i++) {
669            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
670                return true;
671            }
672        }
673        return false;
674    }
675
676    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
677        for (var i = 0; i < digits.length - 1; i++) {
678            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
679                return true;
680            }
681        }
682        return false;
683    }
684
685    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
686        for (var i = 0; i < digits.length - 1; i++) {
687            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
688                return true;
689            }
690        }
691        return false;
692    }
693
694    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
695        for (var i = 0; i < digits.length - 1; i++) {
696            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
697                return true;
698            }
699        }
700        return false;
701    }
702
703    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
704        for (var i = 0; i < digits.length - 1; i++) {
705            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
706                return true;
707            }
708        }
709        return false;
710    }
711
712    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
713        for (var i = 0; i < digits.length - 1; i++) {
714            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
715                return true;
716            }
717        }
718        return false;
719    }
720
721    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
722        for (var i = 0; i < digits.length - 1; i++) {
723            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
724                return true;
725            }
726        }
727        return false;
728    }
729
730    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
731        for (var i = 0; i < digits.length - 1; i++) {
732            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
733                return true;
734            }
735        }
736        return false;
737    }
738
739    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
740        for (var i = 0; i < digits.length - 1; i++) {
741            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
742                return true;
743            }
744        }
745        return false;
746    }
747
748    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
749        for (var i = 0; i < digits.length - 1; i++) {
750            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
751                return true;
752            }
753        }
754        return false;
755    }
756
757    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
758        for (var i = 0; i < digits.length - 1; i++) {
759            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
760                return true;
761            }
762        }
763        return false;
764    }
765
766    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
767        for (var i = 0; i < digits.length - 1; i++) {
768            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
769                return true;
770            }
771        }
772        return false;
773    }
774
775    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimes(digits) {
776        for (var i = 0; i < digits.length - 1; i++) {
777            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrime(digits[i], digits[i + 1])) {
778                return true;
779            }
780        }
781        return false;
782    }
783
784    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
785        for (var i = 0; i < digits.length - 1; i++) {
786            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
787                return true;
788            }
789        }
790        return false;
791    }
792
793    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinSquares(digits) {
794        for (var i = 0; i < digits.length - 1; i++) {
795            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinSquare(digits[i], digits[i + 1])) {
796                return true;
797            }
798        }
799        return false;
800    }
801
802    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
803        for (var i = 0; i < digits.length - 1; i++) {
804            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
805                return true;
806            }
807        }
808        return false;
809    }
810
811    function containsConsecutiveNonTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinCubes(digits) {
812        for (var i = 0; i < digits.length - 1; i++) {
813            if (!isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinCube(digits[i], digits[i + 1])) {
814                return true;
815            }
816        }
817        return false;
818    }
819
820    function containsConsecutiveTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinPrimesAndTwinFibonacciNumbers(digits) {
821        for (var i = 0; i < digits.length - 1; i++) {
822            if (isTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinPrimeAndTwinFibonacci(digits[i], digits[i + 1])) {
823                return true;
824            }
825        }
826        return false;
827    }
828
829
```



A Anhang

```
44 post(digits)
45 return ((differenceArray[0] == 1 || differenceArray[0] == -1) &&
46     containsRepeatedNumbers(differenceArray))
47
48 function isDirection() {
49     var num = [8, 4, 3];
50     var num_direction = num[1] - num[0];
51     for (var i = 0; i < num.length - 1; i++) {
52         if (num_direction * (num[i + 1] - num[i]) <= 0) {
53             post("false");
54         }
55     }
56     post("true");
57 }
```