Hochschule für Technik
und Wirtschaft Berlin
University of Applied Sciences

Untersuchung der Algorithmen und Prozesse der Standortanalyse im Kontext einer Filialplanung eines Einzelhändlers

Masterarbeit

Name des Studiengangs

Internationale Medieninformatik

Fachbereich 4

vorgelegt von

Moritz Thomas

s0544877

Datum:

Berlin, 01.09.2020

Erstgutachter_in: Prof. Dr. Tobias Lenz

Zweitgutachter_in: Sumit Kapoor

Eigenständigkeitserklärung

Hiermit versichere ich, dass ich die vorliegende Masterarbeit selbstständig und nur unter Verwendung der angegebenen Quellen und Hilfsmittel verfasst habe. Die Arbeit wurde bisher in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Berlin, den xx.xx.xxxx

Moritz Thomas

Abstract

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Einführung

Im Jahr 2019 gaben private Haushalte in Deutschland rund 197,3 Milliarden Euro für Lebensmittel aus[1]. Dies verteilt sich auf rund 34.947 Geschäfte zwar kontrollieren die vier Unternehmen Edeka, Rewe, Aldi und die Schwarz-Gruppe rund 70 Prozent des Marktes, dennoch herrscht ein Konkurrenzkampf der jeden noch so kleinen Vorteil gegenüber den Wettbewerbern in direkten Umsatz, Marktanteil oder Kundenzufriedenheit Anstieg nieder spiegelt[2]. Das Spielfeld ist hierbei vielfältig und wechselhaft, Faktoren ändern sich. In Zeiten der fortgeschrittenen Digitalisierung sind Unternehmen mittlerweile auf technische Unterstützung angewiesen, um nicht abgehängt zu werden. Sämtliche Bereiche der Maschinerie Lebensmitteleinzelhandel (im Folgenden LEH) sind teilweise vollständig oder zu großen Teilen von technischen Prozessen durchzogen. Dennoch gibt es LEH bereits seit Jahrzehnten und gängige Prozesse haben sich etabliert und verbreitet. So auch die Standortanalyse und die Filialplanung. Eine spannende Aufgabe der Unternehmen und der Wissenschaft ist es nun neue Technologien für die digitale Umsetzung der gegebenen Prozesse zu nutzen und durch das Zusammenspiel Optimierung und Innovation zu erreichen.

[1] *Konsumausgaben in Deutschland für Nahrungsmittel, Getränke und Tabakwaren bis 2019* 2020.
[2] *Lebensmittel-Discounter in Deutschland bis 2018* 2020.

1.2 Motivation

Viele der in der Standortanalyse verwendeten Prozesse und Algorithmen sind geo-mathematischer oder geo-informatischer Natur und liegen teils komplexer Berechnungen zu Grunde. Diese zu erkunden und ergründen haben sich eigene Wissenschaftsbereiche aus der Mathematik, Betriebswirtschaftslehre, Informatik und dem Ingenieurswesen gebildet, die diverse Studiengänge und Ausbildungen beheimaten. Um diese komplexen Themen zu vereinfachen und den Mitarbeitern des LEH die tägliche Arbeit zu erleichtern, können digitale Prozesse eingesetzt werden. In benutzerfreundlichen, einfach zu verstehenden und visuell ansprechenden Anwendungen sollen sich die Algorithmen und Prozesse der Standortanalyse und Filialplanung verbergen. Solche Anwendungen werden Geo-Informationssysteme (kurz und im Folgenden GIS) genannt. Auf dem Markt existieren derer bereits einige. Big Player sind zum Beispiel Pitney Bowes, ESRI, oder Autodesk. Ebenso existieren einige OpenSource Angebote wie zum Beispiel GRASS GIS oder QGIS. In der Praxis benötigen die Unternehmen dennoch oft Individuallösungen, die entweder auf bestehender Software aufbauen oder diese integrieren. Ziel dieser Arbeit ist es daher eine Anwendung zu entwickeln, die Algorithmen und Prozesse der Standortanalyse und Filialplanung einfach implementiert und in eine solche Individuallösung integriert werden kann. Zu dieser Anwendung gehören also lediglich eine Karte, Karten-Werkzeuge und Geo-Objekte als Teilobjekte eines kompletten GIS.

1.3 Abgrenzung

In dieser Arbeit werden Themenfelder des Geo-Marketings, der Geo-Informatik und -Mathematik sowie diverse Technologien behandelt. Die resultierende Anwendung soll nichts weiter als ein Prototyp darstellen und ist keinesfalls ein komplettes GIS. Vielmehr wird sich auf die Ausarbeitung des Huff-Models zu Standortanalyse und Filialplanung konzentriert mit dem Fokus auf Umsetzbarkeit innerhalb einer Web-Anwendung mit OpenSource Technologien.

Kapitel 2

Grundlagen

Zu den Grundlagen dieser Arbeit zählen Theorien und Konzepte aus dem Geomarketing und der Geoinformatik sowie die angewendeten Technologien im Prototyp. Die folgenden Kapitel stellen die wichtigsten Informationen bereit, die ein allgemeines Verständnis der Anwendung ermöglichen.

2.1 Geomarketing

Aus dem Handbuch Geomarketing von Michael Herter: Geomarketing analysiert aktuelle wie potenzielle Märkte nach räumlichen Strukturen, um den Absatz von Produkten effektiver planen und messbar steuern zu können[1]. Ergänzend befasst sich Geomarketing mit der Beschreibung, Analyse und dem Vergleich beliebiger Märkte und Standorte hinsichtlich ökonomischer Charakteristiken und Potenziale durch Referenzierung und flächendeckende Berechnung von Marktdaten auf geographische Strukturen[2]. Kurz gesagt beschreibt Geomarketing also sämtliche Aspekte des Marketings, die geografischen Bezug haben.

Zu relevanten Themenbereichen des Geomarketings für diese Arbeit gehören Standortanalyse /-planung, Filialplanung sowie die Gravitationsanalyse und das Konzept des Gravitationsmodels. Weiterhin werden die Begriffe Geodaten und Marktdaten erläutert.

[1] Herter und Mühlbauer 2018.
[2] *Definition Geomarketing | Geomarketing* 2021.

2.1.1 Standortanalyse

Allgemein erörtert die Standortanalyse Beschreibung, Untersuchung und Unterscheidung von guten und schlechten IST-Standorten sowie potenziellen neuen Standorten und deren Umfeld hinsichtlich der Eignung für den Absatz bestimmter Produkte[3].

Für die Betrachtung von potenziellen neuen Standorten werden unternehmensinterne Daten (falls vorhanden) und externe Daten anhand von verschiedenen Standortfaktoren untersucht, mit dem Ziel sämtliche potenzielle Standorte auf möglichst wenig Alternativen zu begrenzen und schließlich analytisch die Beste zu bestimmen. Die Unterscheidung kann hierbei in höchster Ebene in vier Kategorien erfolgen[4]:

Zugehörigkeit zur Leistungserstellung beinhaltet Faktoren zur Beschaffung, Produktion und zum Absatz.

Grad der monetären Quantifizierbarkeit beinhaltet harte und weiche Standortfaktoren. Harte Standortfaktoren sind immer quantifizierbar und dienen daher als Grundlage für wirtschaftliche Berechnungen und Kosten. Der Einfluss weicher Standortfaktoren kann nicht eindeutig bestimmt werden und kann mit der selektiven Clusterung all der Faktoren beschrieben werden, die auf dem individuellen Raumempfinden der Menschen in ihrer Lebens- und Arbeitswelt basieren.

Maßstabsebene beschreibt Faktoren der Makro-, Meso- und Mikroebene oder anders der Länder-, Region- und Gemeindeebene.

Grad der Spezifität beschreibt Sektor- und Branchenspezifische Faktoren.

Die Abbildung 2.1 bietet hierzu eine detaillierte Auflistung der Kategorien mit Beispielen.

[3] *Standortanalysen | Geomarketing* 2021.
[4] Haas 2021.

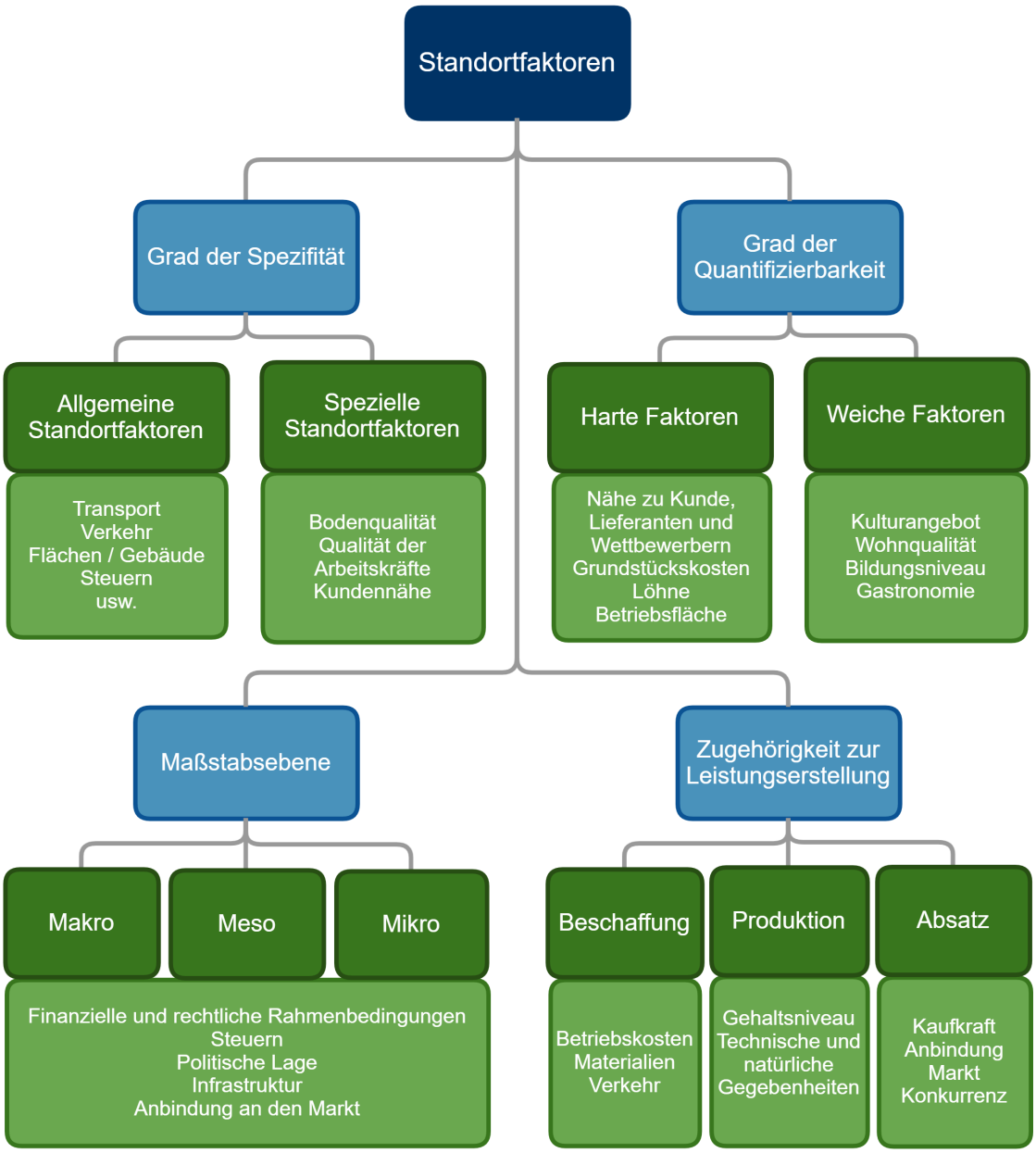

Abbildung 2.1: Eigene Darstellung Standortfaktoren

Weitere zusätzlich relevante Aspekte können sein:

- Zielgruppe - Wer sind die Kunden? Ist das Verkaufsmodell B2C oder B2B?
- Demografische Merkmale - Alter, Geschlecht und Wohnort der Kunden
- Sozioökonomische Betrachtung - Gibt es einen Zusammenhang zwischen Beruf, Bildungsstand oder Einkommen und dem Kaufverhalten?
- Psychografische Merkmale - Hat der Lebensstil, Werte, Motivation oder Ähnliches Einfluss auf das Kaufverhalten?
- Wettbewerbsdichte
- Preispolitik der Wettbewerber
- Größe des Einzugsgebiets
- Kaufkraftwerte des Gebiets
- Erreichbarkeit
- Mietpreise
- Attraktivität des Standortes

Obwohl Filialen geographisch sowie aus betriebswirtschaftlicher Sicht Unternehmensstandorte sind, kann zwischen Filialplanung und Standortplanung unterschieden werden und dementsprechend werden andere Faktoren bei der Standortanalyse wichtiger oder hinzugezogen. Für die Filialplanung zusätzlich relevant werden folgende Faktoren:

- Bekanntheitsgrad, Markenstärke - Was zeichnet die eigene Marke aus? Was setzt sie ab?
- Freies Potential - Herrscht genug Nachfrage?
- Kannibalisierungs-Effekte - Ist der Standort zu nah an anderen, bestehenden Filialen?
- Logistikkosten - Kann ich Lieferwege und Lagerstandorte optimal nutzen?

Für eine einfache mathematische Standortanalyse können nun zunächst die Kategorien anhand der eigenen Marke als mehr oder weniger relevant bewertet und gewichtet werden. Außerdem müssen die einzelnen Kategorien für jeden Standort bewertet werden. Die Bewertungen der Kategorien müssen nun mit der Gewichtung der Kategorie multipliziert werden, um die genaue Faktorbewertung des Standorts für jede Kategorie zu erhalten. Die Faktorbewertungen summiert ergeben nun Standortbewertung. Die Standortbewertungen gegenübergestellt stellen die Standortanalyse abschließend gelistet dar.

2.1.2 Standortplanung/ Filialplanung

Bereits im vorherigen Kapitel erwähnt, wird zwischen der Standortplanung und der Filialplanung geringfügig unterschieden. So legt die Filialplanung ihren Fokus auf mehrere Standorte in einem Filialnetz zusammengefasst, welches bei der Analyse berücksichtigt werden muss. Wohingegen die Standortplanung sich eher auf einige wenige Standorte konzentriert.

Beide Methoden verwenden jedoch in der vorher durchgeführten Standortanalyse herausgestellte Erkenntnisse, um die potenziellen Standorte einzuordnen und eine Entscheidung der nächsten Filialeröffnung zu treffen.

2.1.3 Geodaten

Geodaten sind strukturierte codierte Angaben zur quantitativen und qualitativen Beschreibung von natürlichen oder definierten Objekten der realen Welt. Geodaten vermessen die Welt und beschreiben geographische (Teil-)räume und Orte[5].

Geodaten werden in mehrere Typen klassifiziert und unterliegen definierten Standards, die sich am Markt durchgesetzt haben[6].

So definiert die ISO 19115 einen Standard zur Beschreibung geographischer Informationen anhand von Metadaten[7]. Wichtigste standardisierte Dienste sind unter Anderem:

[5] *Geodaten | Geomarketing* 2021.
[6] *GIS-Standards* 2021.
[7] *ISO 19115* 2021.

- WMS - Web Map Service zum Teilen von Karten
- WFS - Web Feature Service zum Teilen von Feature-Daten

Und zu den wichtigsten standardisierten Datenformaten zählen:

- GeoJSON - JSON Format mit Geo-Spezifika
- KML - Keyhole Markup Language
- GPKG - GeoPackage definiert vom Open Geospatial Consortium

Im Prototyp verwendete Formate sind GeoJSON für Features sowie einfache Bilder (PNG) der Hintergrundkarte.

2.1.4 Marktdaten

Marktdaten oder Marktinformationen beschreiben die individuelle regionale Charakteristik geographischer Gebiete oder Standorte mittels qualitativer Merkmale[8].

Es gibt unternehmensinterne sowie externe Marktdaten. Marktdaten können aus einem internen CRM-System stammen und über Umfragen, Marktforschung, Erhebungen als Primärdaten selbst erhoben werden oder aus staatlichen Statistiken und Branchen- und Wirtschaftsverbänden als Sekundärdaten eingeholt werden.

Im Prototyp verwendete Marktdaten sind aus Sekundärdaten erstellte Beispieldaten für Filialen und Gebiete.

2.2 Geoinformationssysteme

Geoinformationssysteme, kurz GIS, sind Informationssysteme zur Erfassung, Bearbeitung, Organisation, Analyse und Präsentation räumlicher Daten[9]. Ähnlich anderer Informationssysteme besteht ein GIS aus Hardware (Computer, Server, Drucker etc.), Software mit

[8] *Marktdaten | Geomarketing* 2021.
[9] *Geographische Informationssysteme (GIS) | Geomarketing* 2021.

Analyse-Tools (Zeichnen, Kalkulation) und räumlichen Daten (Koordinatensystem, Karten, Geometrien etc.). Gegebenenfalls wird die Liste um eine Verwaltungsebene ergänzt sobald die Daten und Funktionen des GIS Rollen und Rechte behaftet sind. Erste GIS Systeme stammen aus den sechziger Jahren (Canada Geographic Information System[10]). Zu den ersten Nutzern der Systeme gehörten vor Allem Behörden und Universitäten, so wurden viele grundlegende theoretische Konzepte an der Harvard University von Professor Howard Fisher aufgestellt[11]. Mittlerweile haben sich viele Web-GIS etabliert, hierbei wird das Informationssystem über eine Website veröffentlicht oder benutzt. Zu den größten Vertretern moderne Web-GIS zählen vor Allem Google Maps, Bing Maps, OpenStreetMaps als OpenSource-Alternative oder etwas kleinere Anbieter wie HERE oder Yandex.Maps. Kommerzielle Web-Gis bieten meist ein eingeschränktes Funktions-Set, was sie nicht als Produkt für individuelle Software-Lösungen für Firmen in Frage kommen lässt daher greifen viele Firmen auf kommerzielle Desktop-GIS zurück oder lassen sich ganz individuelle Systeme bauen, die aus Web- und Desktop-GIS bestehen. Zu den bekanntesten GIS zählen Produkte von ESRI, Autodesk, Pitney Bowes oder CAIGOS.

2.2.1 Koordinatensysteme und Projektion

In der Geophysik wird die Erde nicht als Kugel sondern Ellipsoid bezeichnet[12]. Wenn man jedoch eine Karte elektronisch auf einem Bildschirm betrachtet, dann sieht man ein Rechteck und keine Anzeichen der Krümmung eines Globus. Die Problematik der Projektion der Erde auf eine flache Darstellung hat über den Lauf der Jahre mehrere Lösungen produziert, die jedoch alle mit Koordinatensystemen und Projektionsrechnungen zu tun haben, und bietet Stoff für ein eigenes Wissenschaftsfeld. Aufgrund dessen versuchen die nächsten Zeilen, die Problematik und Anwendung im Prototyp kurz zu beschreiben.

In einem GIS muss die Projektion zwangsläufig enthalten sein und das bestenfalls für mehrere Projektionstypen. Die meisten Web-Karten wie Google Maps oder Bing Maps benutzen die Projektion World Geodetic System 1984 (kurz WGS 84) mit dem Koordinatenreferenzsystem (engl. CRS) der European Petroleum Survey Group (kurz EPSG)

[10] *Origins of the Canada Geographic Information System | ArcNews* 2020.
[11] *Untitled Document* 2005.
[12] Jung 1956.

4326, welches Koordinaten in Grad darstellt[13]. OpenLayers hingegen benutzt standardmäßig WGS 84 mit EPSG 3857, welches Koordinaten in Metern zur Ursprungskoordinate darstellt[14]. Die Abbildung 2.2 zeigt die Koordinaten eines Punktes in Berlin in beiden Koordinatenreferenzsystemen.

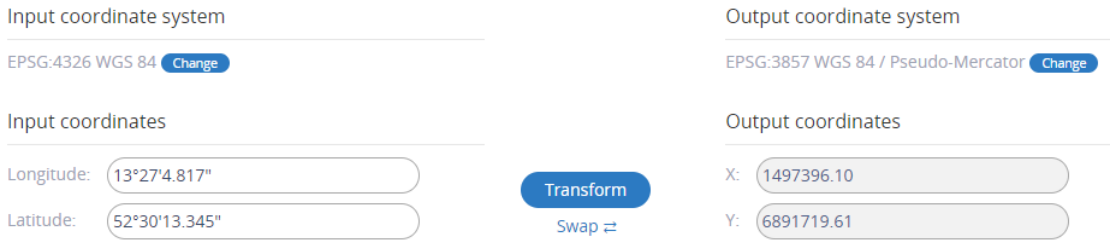

Abbildung 2.2: Bildschirmaufnahme der Transformation von EPSG 4326 zu EPSG 3857 von epsg.io[15]

Im wesentlichen werden im Prototypen die oben beschriebenen Projektionen verwendet. Die Geodaten der Filialen sowie der Zensusgebiete sind in EPSG 4326 erfasst und müssen dementsprechend vor der Darstellung transformiert werden. Entsprechende Funktionen sind im Kapitel 4 Implementierung beschrieben.

2.3 Genutzte Technologie

Für die prototypische Ausarbeitung des Modells wurde auf modernste Technologien und Frameworks zugegriffen. So wird als Karten-Framework /gls(ol) in der Version 6.4.3 verwendet sowie /gls(angular) in der Version 10. Die Darstellung 2.3 zeigt die genutzten Technologien als Grafik.

[13] *WGS 84 - WGS84 - World Geodetic System 1984, used in GPS - EPSG* 2021.
[14] *WGS 84 / Pseudo-Mercator - Spherical Mercator, Google Maps, OpenStreetMap, Bing, ArcGIS, ESRI - EPSG* 2021.

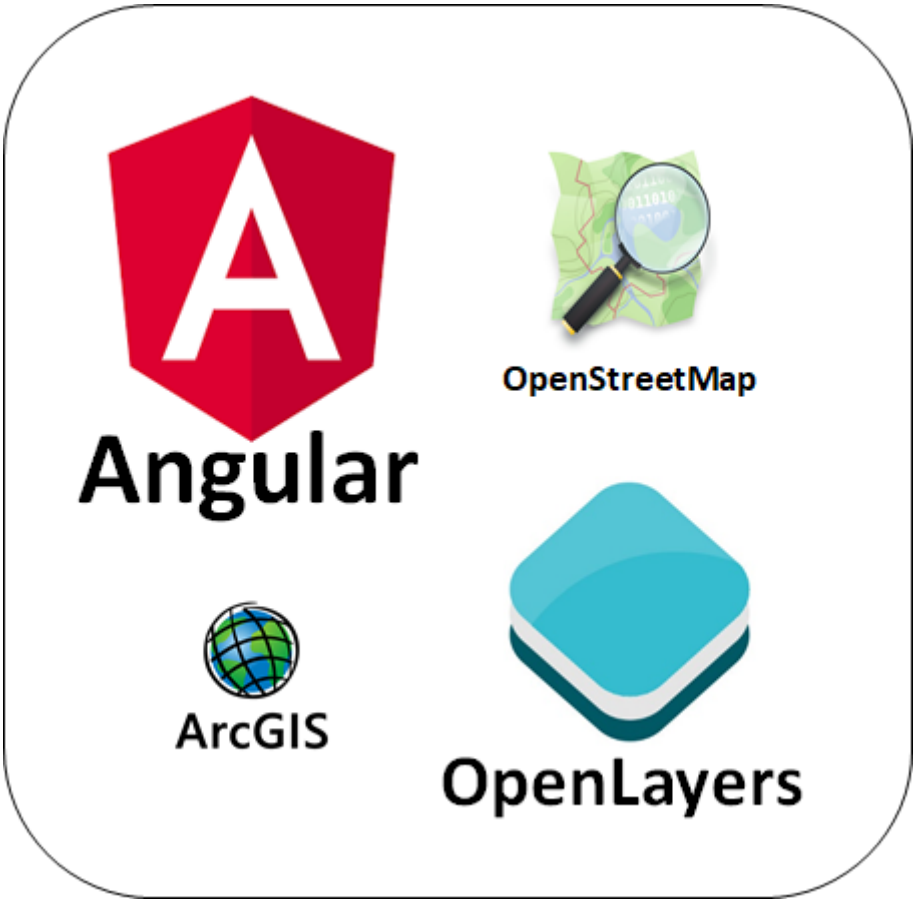

Abbildung 2.3: Eigene Abbildung des Technologie Stacks der Anwendung

OpenLayers bietet ein breites Portfolio an Funktionen und Features passend für sämtliche Anforderungen realer Geo-Informationssysteme die in der Praxis verwendet werden. Im Bezug auf die prototypische Anwendung dieser Arbeit sind vor Allem die Unterstützung der Darstellung von Punkten (Filialen) und der Gravitations-Gebiete in GeoJSON-Format auf verschiedenen Layern notwendig sowie die einfache Konfigurierbarkeit von Karten-Interaktion wie zum Beispiel das Platzieren, Verschieben und Editieren von Punkt-Objekten (Filialen) auf der Karte.

Weiterführend soll die Anwendung schnell, kompakt und modern sein, um die Relevanz aus Performance Gründen auf dem Markt gewährleisten zu können. Daher erfolgt die Umsetzung mit dem auf TypeScript basierenden Framework Angular. Besonderes Augenmerk liegt hierbei im Prototypen eigentlich nur auf der simplen, kompakten und schnellen Auslieferung eines Web-Servers als Host der Anwendung. Zu den für die weiterführende

Entwicklung relevant werdenden Features des Frameworks zählen eine große Community, einen fortlaufenden Support sowie eine fortlaufende Entwicklung durch Google, die Verwendung von TypeScript einem Superset von JavaScript mit Verbesserter Funktionalität und sämtlichen Features, die für Entwicklung einer effizienten und anspruchsvollen Single-Page-Webanwendung benötigt werden.

Kapitel 3

Konzept

Die Algorithmen und Prozesse einer Filialplanung spiegeln sehr gut einen allgemeinen Anwendungsfall moderner GIS. Neben einer einfachen Hintergrundkarte sind die Mitarbeiter auf Zeichenwerkzeuge, Geo-Daten-Anzeige, thematische Karten und Kennzahlen Berechnungen angewiesen, um akkurate und fundierte Prognosen und Planungen treffen zu können. Viele theoretische Konzepte und Prozesse passieren hierbei im Hintergrund und sind nicht direkt ersichtlich.

In den folgenden Kapiteln werden Prozesse der Standortanalyse beschrieben und der Fokus auf das Gravitationsmodell von Huff als Algorithmus zur Bestimmung des Marktanteils gelegt. Ebenso wird die daraus entstehende Architektur des Prototyps vorgestellt.

3.1 Algorithmen und Prozesse

Die Filialplanung ist ein Prozess, der sich aus mehreren Teilprozessen zusammensetzt. Einer dieser Teilprozesse ist die Standortanalyse bezogen auf Filialen. Die Wahl eines Standortes ist keinesfalls immer eindeutig und anhand des potenziellen Umsatzes der neuen Filiale zu erkennen. So kann es zum Beispiel durchaus sinnvoll sein eine Filiale trotz möglicher Kannibalisierung anderer eigener Filialen dennoch zu eröffnen, weil dies gleichzeitig zu Umsatzeinbußen bei Konkurrenzfilialen führt. Der eigene Marktanteil und Gesamtumsatz ist der entscheidende Faktor bei der Filialplanung. Durch die Anwendung ausgewählter Algorithmen gilt es also folgende Leitfrage zu beantworten:

Wo befindet sich der ideale Standort, um eine neue Filiale zu eröffnen?

Um diese Frage fundiert beantworten zu können, ergeben sich mehrere vorliegende Fragen:

Wo kann ich überhaupt eine neue Filiale eröffnen?

Was ist mit meinen Konkurrenten, was ist mit Kannibalisierung meiner eigenen Filialen?

Wie kann ich folglich also sicherstellen, dass sich mein Gesamtumsatz durch die Neueröffnung steigert?

Die Suche nach dem idealen Standort beginnt zunächst bei der Eingrenzung möglicher Standorte. Im modernen Stadtbild ist die Anzahl freier Standorte begrenzt. Meist bestimmt das Angebot die Möglichkeiten.

Existiert nun eine Liste möglicher Standorte, muss für jeden Standort der Einfluss auf das bestehende Filialnetz erfasst werden. Das bestehende Netz kann nur eigene Filialen beinhalten, um die Kannibalisierung meines eigenen Netzes durch eine Neueröffnung zu analysieren, oder Konkurrenzfilialen und Lager einschliessen, um den Einfluss auf den Gesamtmarkt zu berechnen.

Wichtige Parameter für die Berechnung des Marktanteils der Filialen im Netz sind Attraktivität, Nähe zu Zielgruppen und die Entfernung zu anderen Filialen.

Über das Huff-Modell kann die Marktanteil-Auswirkung aller Standorte berechnet werden. Im Folgenden wird das Modell und zugehörige Komponenten im Detail vorgestellt.

Um eine realistische Abbildung der Welt auf eine Karte zu bringen benötigt es eine Umrechnung eines Ovales, quasi der Erdkugel, auf eine Rechteck, im Fall eines Web-Gis des Bildschirms.

3.1.1 Gravitationsmodell

Das *Huff-Modell* (engl. Huff Gravitation Model) ist ein mathematisches Modell zur Abgrenzung und Segmentierung von Marktgebieten **Roy2004**. Das Modell bestimmt die

$$P_{ij} = \frac{A_j^{\alpha} D_{ij}^{-\beta}}{\sum_{j=1}^{n} A_j^{\alpha} D_{ij}^{-\beta}}$$

Abbildung 3.1: Formel des Huff-Modells

Wahrscheinlichkeit, mit der Kunden einen Standort (Filiale, Einkaufszentrum) in Abhängigkeit von Distanz und Attraktivität aufsuchen. Die Formel wird im allgemeinen wie folgt dargestellt:

Die Wahrscheinlichkeiten werden nun auf der Karte als Punkte dargestellt, die rund um den Standort platziert werden. Die Punkte der selben Wahrscheinlichkeiten werden zu Isowahrscheinlichkeitslinien verbunden. So entstehen zu jedem Standort verschiedene Gravitationsebenen, die farblich gekennzeichnet werden. Die Beeinflussung der verschiedenen Gravitationsebenen führt zu mehreren Farbverläufen, die ein komplexes Bild der Standort Landschaft bilden. In seiner einfachsten Form berücksichtigt das Modell nur die Distanz und eine simple Kennzahl der Attraktivität (zum Beispiel in Form eines Rankings) für die Wahrscheinlichkeitsberechnung. Hierzu wird zunächst ein maximaler Einflussbereich der Filialstandorte definiert. Dieser Bereich stellt die Grundlage der Berechnungen dar und muss deswegen bekannt und mit berechenbaren Kennzahlen gefüllt sein. Zur Bestimmung des Bereiches können einfache geografische Abstände oder Gebiete benutzt werden, wie zum Beispiel eine Berechnung auf Grundlage Berlins als Einflussbereichs. Vor Allem aber kommen zeitliche oder örtliche Parameter zum Einsatz. So macht es aus wirtschaftlicher Sicht viel mehr Sinn das Gebiet anhand von Fahrzeitzonen zu berechnen. So würde ein potenzieller Kunde aus Brandenburg wahrscheinlich auch in einer Filiale in Berlin einkaufen gehen, wenn diese attraktiver, örtlich näher oder besser zu erreichen ist. Berechnet wird also ein Einzugsbereich, um die Filiale herum. Ob dies nun eine maximale Fahrtzeit von 30 Minuten ist oder eher eine maximale Distanz von 30 Kilometern, ist auf die einzelne Filiale oder den gewählten Standard der Berechnung zurückzuführen.

Um die Huff-Formel korrekt kalibrieren zu können, genauer gesagt die Parameter alpha und beta empirisch bestimmen zu können, müssen folgende Schritte befolgt werden:

- Abgrenzung des Erhebungsgebiets

- Unterteilung des Erhebungsgebiets in Untergebiete
- Zentroiden der Gebiete festlegen
- Alle konkurrierende Einrichtungen identifizieren und Koordinaten erfassen
- Entfernungen zwischen den Zentroiden aller Gebiete und aller Einrichtungen berechnen
- Spezifizieren aller Eigenschaften zur Kundenbeeinflussung
- Wirtschaftliche, soziale und demografische Daten für Gebiete angeben
- Studie durchführen für die Frequenz in welcher Kunden Einrichtungen besuchen

Weiterführend muss bestimmt werden wie und ob sich das Potenzial über den Verlauf der Distanz des Einzugsbereiches verändert. Bleibt das Potenzial konstant, würde dies bedeuten die Kunden in den äußeren Bereichen des Gebiets kommen mit der gleichen Wahrscheinlichkeit in die Filiale wie die Kunden in den unmittelbar angrenzenden Bereichen. Aber gegenteilig wäre ein linear abnehmendes Potenzial wahrscheinlich ebenso nicht vollständig realitätsgetreu, da Kunden ab einer Distanz, die zu groß für den Fuß-Weg wäre, eher das Auto oder den öffentlichen Nahverkehr wählen und dann eventuell direkt zu einer attraktiveren Filiale weiter weg fahren würden. Daher kann als grundlegende Distanzfunktion quasi eine beliebig komplizierte Formel gewählt werden. Aus Gründen der Vereinfachung und Demonstration wird für den Prototyp eine einfache linear abnehmende Distanzfunktion gewählt.

Nachdem nun zunächst die Potenzialberechnung für eine einzelne Filiale anhand der beschriebenen Parameter und Funktionen erfolgen konnte gilt es nun das Potenzial in einem bestehendem Filialnetz zu berechnen. Als Ergebnis wird hierbei eine Wahrscheinlichkeitsberechnung für sämtliche Filialen des Netzes erwartet, sodass jedem Feld des summierten Gesamt-Einzugsgebietes einen Wahrscheinlichkeitswert zugeordnet werden kann, der beeinflusst von sämtlichen anderen Filialen des Netzes, für jede Filiale des Netzes unterschiedlich sein kann und dementsprechend eingefärbt werden kann. Das Endergebnis zeigt somit die beschriebene farbliche Gravitationskarte.

3.2 Architektur

Die Architektur der Anwendung ergibt sich hauptsächlich aus den fachlichen Anforderungen und technischen Voraussetzungen.

Das Diagramm 3.2 zeigt die Komponenten der Anwendung im Zusammenspiel in einer eigenen Darstellung.

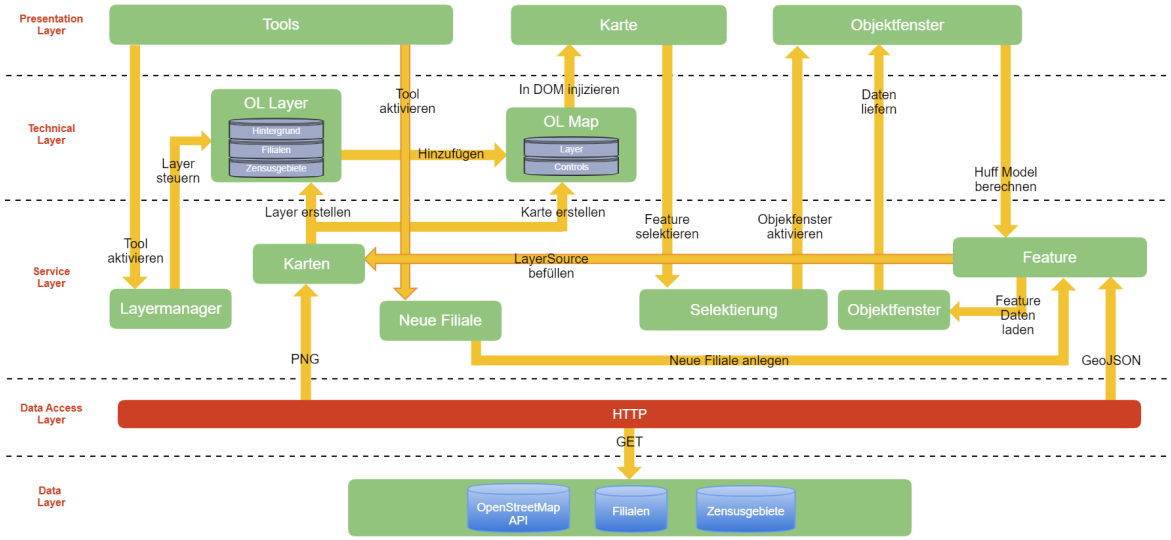

Abbildung 3.2: Komponentendiagramm der Anwendung in eigener Darstellung

Die Anwendung baut im wesentlichen auf vier Komponentengruppen auf:

- Angular Komponenten
- Angular Services
- OpenLayers Karte und Layer
- Geo-Daten

OpenLayers als GIS-Software bildet hierbei den Kern in Form der Karte. Die Karte bietet an sich bereits einen großen Teil der GIS-Werkzeuge, die für die Anwendung benötigt werden. Die Karte stellt die Layer und deren Sourcen in der richtigen Projektion dar und bietet die Selektierungs-Interaktion der einzelnen Features. Sie sorgt also dafür, dass die

ovale Weltkarte auf den flachen Bildschirm projiziert wird und der Klick auf eine Pixel-Koordinate des Bildschirmes umgerechnet eine geografische Koordinate widerspiegelt. Die Karte wird in der Anwendung im BaseMap-Service mit Parametern erstellt und in der OpenLayersMap-Komponente in den DOM injiziert.

Der Karte werden drei Layer hinzugefügt:

- Hintergrund-Layer
- Filial-Layer
- Zensusgebiete-Layer

Fachlich betrachtet kann der Hintergrund-Layer auch als Hintergrundkarte bezeichnet werden, technisch ist er jedoch ebenso ein Layer wie die Filialen und Gebiete. Die Layer werden im BaseMap-Service erstellt und der Karte in der OpenLayersMap-Komponente hinzugefügt. Zu den Layern gehören entsprechende Sources, über die Features in den Layer und somit dargestellt werden können.

Der Hintergrund-Layer ist ein Kachel-Layer (Englisch: Tile-Layer). Der Layer wird durch ein Gitter (Englisch: Grid) in einzelne Kacheln eingeteilt, die alle einzeln mit Bildern befüllt werden. Dies ist ein Optimierungsschritt, der den Datentransfer reduziert sowie die Renderingzeit und -performance verbessert. Die Kachelbilder werden über Http-Requests an die OpenStreetMap-API geladen.

Die beiden Feature-Layer der Filialen und Gebiete sind Vektor-Layer (Englisch: Vector-Layer). Die jeweiligen Sources der Layer lesen die Features über lokale GeoJSON-Dateien im Feature-Service ein und befüllen den Layer im BaseMap-Service mit den Sourcen.

Nun sind die Features über ihre Layer auf der Karte zu sehen und können über eine Selektions-Interaktion in der Karte ausgewählt werden. Über den SelectFeature-Service wird daraufhin das Objektfenster aktiviert, welches Informationen über das Feature enthält und die Berechnung des Huff-Modells für die selektierte Filiale starten kann. Die Daten des Features werden über den Objektfenster-Service in das Objektfenster injiziert.

Über die Tool-Komponente kann eine neue Filiale auf die Karte gesetzt werden und die Layer gesteuert werden.

Sobald eine neue Filiale auf der Karte gesetzt ist wird über den Neue-Filiale-Service ein weiteres Feature in die Filiale-LayerSource geladen und über das Objektfenster kann nun die Berechnung des Huff-Modells gestartet werden.

Über das LayerManager-Tool kann im LayerManager-Service die Sichtbarkeit und Reihenfolge der Layer gesteuert werden.

Kapitel 4

Implementierung

Das Kapitel Implementierung erfasst die technische Dokumentation des erarbeiteten Prototyps. Im Detail werden in den folgenden Seiten die technische Umsetzung des in Kapitel 3 entworfenen Konzepts beschrieben. Beschriebene Funktionalitäten werden mit Bildausschnitten unterstützt.

4.1 Prototyp

Die Anwendung wurde als Angular Projekt mittels der Angular CLI erstellt. Über den CLI Befehl

Codeauszug 4.1: Erstellen eines neuen Projektes

```
1 $ ng new gravitationsmodel
```

generiert die CLI ein kompilierbares und ausführbares Angular Projekt mit essentiellen Abhängigkeiten und Strukturen. Angular verwendet standardmäßig NPM als package manager. Eine *package.json*, welche sämtliche Abhängigkeiten dokumentiert, wird bereits mit erstellt. Um das Setup abzuschließen müssen weitere Abhängigkeiten in Form von *npm packages* installiert werden. Über den Befehl

Codeauszug 4.2: Hinzufügen des OpenLayers Pakets

```
1 $ ng add ol
```

fügt die CLI automatisch das Paket von OpenLayers hinzu und für die korrekte Typisierung in TypeScript das notwendige *types* Paket für OpenLayers hinzu.

Nachdem die technischen Voraussetzungen geschaffen sind kann mit der Implementierung begonnen werden. Auch hierbei bietet die CLI Unterstützung in Form von *Scaffolding* Befehlen.

Codeauszug 4.3: Angular schematic Befehl zum Erstellen

```
1 $ ng generate <schematic> [name]
```

generiert Komponenten, Services, Models, Klassen und weitere Code-Gerüste durch den passenden Präfix-Parameter und Namen. Angular Komponenten bestehen meist aus einer TypeScript-Klasse (*.component.ts*), einem Template (*.component.html*) sowie Dateien für Styling (*.css* oder *.scss*) und Tests (*.component.spec.ts*).

Codeauszug 4.4: Erstellen der Objektfensterkomponente

```
1 $ ng generate component objektfenster
```

erstellt *objektfenster.component.ts*, *objektfenster.component.html* sowie *objektfenster.scss* und *objektfenster.component.spec.ts* Dateien.

Mit Hilfe der CLI lassen sich so die Code-Gerüste schnell erstellen und die CLI übernimmt sogar die in Angular nötige *Dependency Injection* indem neue Komponenten direkt in der entsprechenden *.module.ts* deklariert werden[1].

Ebenfalls im Zuge der Projekterstellung über die CLI werden Konfigurationsdateien angelegt, die wichtige Einstellungen zum Kompilieren, Bau, Start und Hosten (engl. host) der Anwendung enthalten.

Die Anwendung kann nun über

Codeauszug 4.5: Starten der Anwendung

```
1   $ ng serve
```

[1] *Angular CLI* 2021.

gestartet werden. Der Befehl kompiliert, baut und startet die Anwendung über einen lokalen Webserver. Über *localhost:4200* kann die Anwendung im Browser aufgerufen werden.

Der Kern der Anwendung ist eine Karte mit der interagiert wird. Um diese darzustellen, muss zunächst ein OpenLayers Map-Objekt erstellt und in den DOM eingehängt werden[2]. Dies geschieht in der OpenLayersMap-Komponente und dem BaseMap-Service. Über das bereits installierte OpenLayers Paket wird das Map-Objekt importiert und initialisiert.

Codeauszug 4.6: Erstellung der Karte

```
1
2   const targetId = 'map';
3   const longitude = 13.451338;
4   const latitude = 52.503707;
5   const zoomLevel = 11;
6
7   const coordinate = fromLonLat([longitude, latitude]);
8
9   const view = new View({
10    center: coordinate,
11    zoom: zoomLevel
12  });
13
14  const map = new Map({
15    target: targetId,
16    layers: [],
17    view,
18  });
```

Die Karte wird mit der notwendigen Id des HTML-Elements, in welchem die Karte angezeigt werden soll, und einem View initialisiert. Das View-Objekt besteht aus einer Center-Koordinate und der initialen Zoomstufe und stellt den Begrenzungsrahmen (engl. Bounding Box) der Karte dar. Zu beachten ist ebenfalls hierbei die Transformation der Koordinaten von EPSG:4326 in das in OpenLayers standardmäßig genutzte EPSG:3857

[2] *OpenLayers v6.5.0 API - Class: Map* 2021.

über die Funktion *fromLonLat*. Die Karte wird ebenfalls mit Standard-Interactions und Standard-Controls initialisiert jedoch sind noch keine Layer hinzugefügt. Somit ist jetzt auch noch nichts auf der Karte zu sehen und lediglich bereits erwähnte Controls und Interactions sind sichtbar und bedienbar. Hierzu zählen:

Controls Zoom

Controls Rotate (Unsichtbar bei Rotation 0)

Controls Attribution

Interactions Drag rotate

Interactions Drag pan

Interactions Drag zoom

Interactions Double click zoom

Interactions Mouse wheel zoom

Interactions Pinch rotate (Touchscreen)

Interactions Pinch zoom (Touchscreen)

Interactions Keyboard pan

Interactions Keyboard zoom

Als nächstes wird der Hintergrundkarten-Layer hinzugefügt. Hierzu wird ein neuer Tilelayer initialisiert, eine neue Layer-Quelle (engl. Source) hinzugefügt und der Layer der Karte hinzugefügt.

Codeauszug 4.7: Erstellung der Hintergrundkarte

```
1
2   const backgroundLayer = new TileLayer();
3   backgroundLayer.setSource(new OSM());
4   backgroundLayer.set('name', 'Hintergrund');
5
6   map.addLayer(backgroundLayer);
```

Der Hintergrundkarten-Layer ist ein Kachel (engl. Tile) Layer, da dies die Datenabfrage an den Kartendienstleister, in diesem Fall OpenStreetMaps, reduziert und eine optimale Render-Performance erzeugt. Ein Tile-Layer teilt den Kartenausschnitt in ein Gitter (engl. Grid) aus vielen kleinen Kacheln ein. Jeder dieser Kacheln setzt ihren eigenen Http-Request an den in der Source angegeben Provider mit den Kachel-Koordinaten und Zoom-Stufe als Parameter ab und bekommt einen Kartenausschnitt in PNG-Format zurück. Die einzelnen Bilder werden dann im Cache des Browsers gespeichert und sorgen somit für verminderten Datentransfer sowie schnelles Laden bei erneutem Aufruf. OpenLayers bietet für OpenStreetMap sowie Bing Maps eigene Source-Objekte, welche für Bing nur noch einen API-Key benötigt. Um den Layer später einmal einfach identifizieren zu können, wird dem Layer noch eine Name-Eigenschaft zugewiesen.

Nach der Hintergrundkarte müssen nun noch Layer für die Filialen sowie die Zensusgebiete implementiert werden. In der OpenLayersMap-Komponente werden dazu zwei neue Layer mit passenden Source-Objekten angelegt.

Codeauszug 4.8: Erstellung der Filial- und Gebietelayer

```
1
2   const filialLayer = new VectorImageLayer({
3     visible: true,
4     zIndex: 2
5   });
6   filialLayer.set('name', 'Filialen');
7   filialLayer.setSource = new VectorSource({
8     format: this.geoJSONFormat,
9     strategy: bbox
10  });
11
12  const gebieteLayer = new VectorImageLayer({
13    visible: true,
14    zIndex: 1
15  });
16  gebieteLayer.set('name', 'Gebiete');
17  gebieteLayer.setSource = new VectorSource({
18    format: this.geoJSONFormat,
19    strategy: bbox
```

```
20  });
21
22  map.addLayer(filialLayer);
23  map.addLayer(gebieteLayer);
```

Bei der Initialisierung der Layer muss die Z-Ebene der Filialen höher sein, da sie später in der Anwendung über den Gebieten liegen sollen und nicht von diesen überlagert werden sollen. Bei VectorSources kann zwischen den Ladestrategien Tile und Bounding Box entschieden werden. Sie bestimmt wann neue Features auf der Karte geladen und angezeigt werden sollen. Bei der Tile-Strategie wird wie bei der Hintergrundkarte der Kartenausschnitt in ein Kachelgitter unterteilt und bei der Bounding Box Strategie wird der gesamte Kartenausschnitt (engl. Bounding Box) gewählt. Bei besonders vielen Daten sollte auf die Tile-Strategie zurückgegriffen werden aber im Prototypen reicht die Bounding Box Strategie vollkommen aus.

OpenLayers definiert standardmäßig für jeden Vector Layer einen Stil, der die einzelnen Features je nach Geometrietyp auf der Karte darstellt. Soll dieser Stil geändert werden, muss auf dem Layer ein neuer Stil definiert werden. Dies kann über ein statisches Style-Objekt erfolgen oder über eine dynamische Style-Funktion.

Codeauszug 4.9: Erstellen des Filialstyles

```
1
2   const filialStyle = new Style({
3     image: new Icon({
4       color: '673ab7'
5     }),
6     text: new Text({
7       fill: new Fill({
8         color: 'FFFFFF'
9       })
10    })
11  });
12
13  const filialStyleFunction = (feature, resolution) => {
14    if (feature.get('selected') === true) {
```

```
15        filialStyle.getImage()
16        .setSource('assets/geometries/icons/PIN_selected.svg');
17      } else {
18        filialStyle.getImage()
19        .setSource('assets/geometries/icons/PIN.svg');
20      }
21      filialStyle.getText().setText(feature.getId().toString());
22      return filialStyle;
23    }
24
25    filialLayer.setStyle(filialStyleFunction);
```

Nun werden realistische Geo- sowie Marktdaten benötigt.
Die Filialen sind einfache Koordinaten im Raum Berlin. Ein Testdatensatz mit 97 Koordinaten wurde über geojson.io [3] angelegt. Die Koordinatenliste wird bereits in GeoJSON angelegt und die Einträge müssen lediglich um eine Id, Anzahl der Parkplätze sowie die Verkaufsfläche der Filiale ergänzt werden.
Ein Testdatensatz mit 1220 Zensusgebieten des Geoportals Berlin[4] wurde über das ArcGIS Hub[5] heruntergeladen. Die Gebiete werden ebenfalls bereits in GeoJSON exportiert. Da es sich jedoch um zu viele Einträge für eine manuelle Ergänzung der Marktdaten handelt, werden Marktdaten für die einzelnen Gebiete beim einlesen der Datei ergänzt.

Die Datensätze werden über das Feature Format GeoJSON in OpenLayers eingelesen und in eine FeatureCollection (bei einzelnen Daten in ein Feature) umgewandelt, welche der jeweiligen Layer Source hinzugefügt wird.

Codeauszug 4.10: Laden der GeoJSON-Dateien der Filialen und Gebiete

```
1
2     this.http.get('../assets/filialen.json').subscribe(value => {
3       const readFeatures = this.geoJSONFormat.readFeatures(value);
4       readFeatures.forEach(feature => {
5         feature.setId(feature.get('id'));
6         feature.set('type', FeatureTypeEnum.FILIALE);
```

[3] geojson.io
[4] *Geoportal / Land Berlin* 2021.
[5] *Verkehrszellen - Berlin* 2021.

```
7       });
8       filialLayerSource.addFeatures(readFeatures);
9     });
10
11    this.http.get('../assets/zensusgebiete.json').subscribe(value => {
12      const readFeatures = geoJSONFormat.readFeatures(value);
13      readFeatures.forEach(feature => {
14        feature.setId(feature.get('id'));
15        feature.set('type', FeatureTypeEnum.ZENSUSGEBIET);
16      });
17      gebieteLayerSource.addFeatures(readFeatures);
18    });
19
20  }
```

Es handelt sich bei den Filial- sowie Gebietsdaten um fiktionale Daten, jedoch sollten diese einen repräsentativen Charakter haben und möglichst nah an realistischen Marktdaten gewählt werden. Als Referenz der Parkplätze und Verkaufsfläche dienen Werte der Statistik Ëntwicklung der durchschnittlichen Verkaufsfläche der Lebensmittel-Discountmärkte Lidl in Deutschland in den Jahren 2009 bis 2019 (in Quadratmetern)"von handelsdaten.de[6] sowie die Anlage zu Nummer 51.11 der Verwaltungsvorschrift zu Landesbauordnung Nordrhein-Westfalen[7]. Aus den Quellen lässt sich eine durchschnittliche Verkaufsfläche von 898 Quadratmetern sowie 1 Parkplatz pro 10 Quadratmetern Verkaufsfläche erschließen. Um den Filialen verschiedene Verkaufsflächen und Parkplätze zuzuordnen wurden Zufallszahlen im Bereich 798 bis 998 für die Verkaufsfläche gewählt sowie die sich daraus ergebende Anzahl der Parkplätze.
Bei den Marktdaten der Gebiete handelt es sich um Angaben zu Anzahl der Einwohner, durchschnittliche Kaufkraft und durchschnittliche Ausgaben für Lebensmittel pro Gebiet. Als Referenz für die Marktdaten dienen Werte der Statistik "Kaufkraft je Einwohner nach Bundesländern im Jahr 2021"von statista.de[8] sowie Werte der Statistik "Konsumausgaben

[6] *Lidl - Verkaufsfläche der Discountmärkte in Deutschland | Zeitreihe | Handelsdaten.de | Statistik-Portal zum Handel* 2021.
[7] *Anlage zu Nr. 51.11 VV BauO NRW Richtzahlen für den Stellplatzbedarf* 2021.
[8] *GfK Kaufkraft je Einwohner nach Bundesländern 2021* 2021.

privater Haushalte in Deutschland"vom Statistischen Bundesamt[9]. Aus den Quellen lässt sich eine durchschnittliche Kaufkraft pro Einwohner pro Monat (kurz DKpEpM) von 1.819 € für Berlin sowie durchschnittliche Ausgaben für Nahrungsmittel pro Monat pro Haushalt (kurz DAfNpMpH) von 356 € für Deutschland erschließen. Aus Gründen der Einfachheit ergeben sich daraus die Kennzahlen pro Gebiet wie folgt:

$$KaufkraftproGebiet = Einwohner * DKpEpM \tag{4.1}$$

$$AusgabenLebensmittelproGebiet = Einwohner * DAfNpMpH \tag{4.2}$$

Da leider keine genauen Angaben der Einwohner pro Zensusgebiet verfügbar sind, wurde die Anzahl der Einwohner anhand der Größe des Gebiets mal dem Faktor 0.0045, welcher empirisch bestimmt wurde, berechnet:

$$EinwohnerproGebiet = GrößeGebiet * 0.0045 \tag{4.3}$$

Nachdem nun Marktdaten und Geodaten erfolgreich simuliert wurden und in der Karte zu sehen sind, werden zunächst die UI-Elemente der Oberfläche implementiert. Damit schlussendlich die Berechnung nach dem Huff-Modell-Algorithmus erfolgen kann, muss die Nutzbarkeit (engl. usability) der Anwendung etabliert werden. Bisher existieren in der Anwendung lediglich eine Karte mit Standard-Interaktionen mit Layern für einen Kartenhintergrund, Filialen und Zensusgebiete.
Implementiert werden folgende UI-Elemente (siehe auch Abbildung 4.1):

- Werkzeug-Set
  - Werkzeug: Neue Filiale anlegen
  - Werkzeug: Layermanager
- Objektfenster für Filialen und Gebiete

[9] *Konsumausgaben privater Haushalte in Deutschland* 2021.

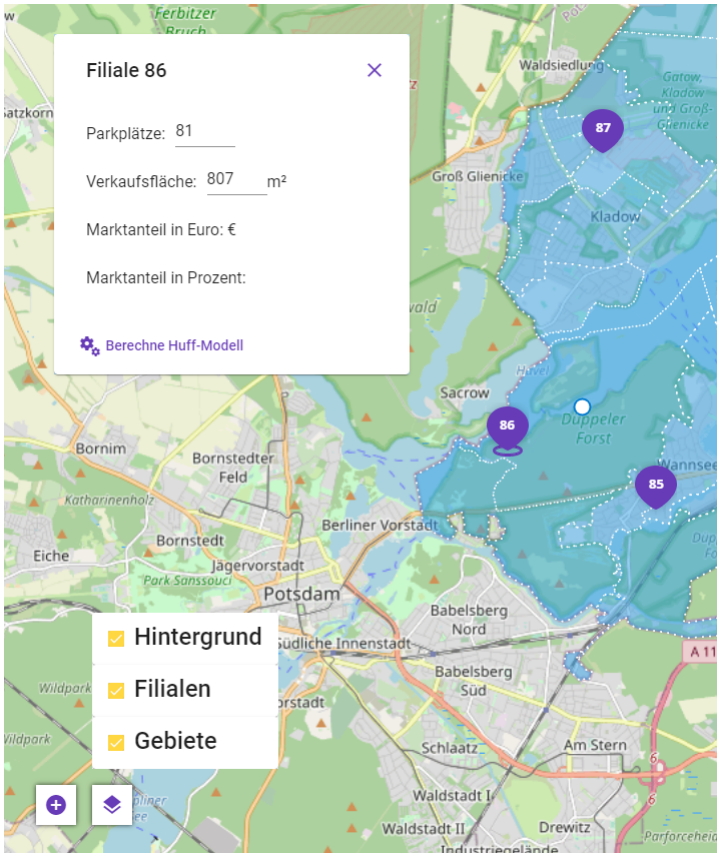

Abbildung 4.1: Bildschirmausschnitt der UI-Elemente

Bei der Umsetzung der UI-Elemente wurde auf die Komponenten-Bibliothek *Angular Material* für fertige UI Komponenten zugegriffen[10]. Zu den benutzen Komponenten zählen Material Button[11], Icon[12], Card[13], List[14], Input[15] sowie Checkbox[16].

Um eine bessere Visualisierung einzelner Featurelayer zu ermöglichen, sollen die Layer einzeln aus- beziehungsweise eingeblendet werden können. Dies geschieht im Werkzeug Layermanager. Hierzu müssen also innerhalb des Werkzeuges alle Layer angezeigt werden und über eine Checkbox die Sichtbarkeit verändert werden können.

[10] *Angular Material* 2021.
[11] *Angular Material Button* 2021.
[12] *Angular Material Icon* 2021.
[13] *Angular Material Card* 2021.
[14] *Angular Material List* 2021.
[15] *Angular Material Input* 2021.
[16] *Angular Material Checkbox* 2021.

Für die Konfiguration des Layermanagers werden zunächst über den BaseMap-Service alle Layer der Karte abgerufen. Aus den Layer-Objekten werden nun Name- und Sichtbarkeit -Eigenschaften in jeweils ein neues LayerManagerEntry-Objekt extrahiert. Im Layermanager können nun der Name und die Sichtbarkeit pro Layer dargestellt werden, wobei die Sichtbarkeit als Checkbox dargestellt wird. Mit der Interaktion einer Checkbox wird über den zugehörigen Namen der entsprechende Layer im BaseMap-Service identifiziert und das *Visible* Attribut angepasst. Der Layermanager erscheint als Fenster nach Aktivierung des entsprechenden Knopfes (Icon: *layer*, siehe Abbildung 4.1).

Codeauszug 4.11: Layermanager

```
1
2  type LayerManagerEntry = { layerIdentifier: LayerIdentifier, visibility: ↩
       boolean };
3
4  public layerManagerEntries: LayerManagerEntry[];
5
6  this.layerManagerEntries = ↩
       baseMapService.getLayers().getArray().map(layer => {
7    return {
8      layerIdentifier: layer.get('name'),
9      visibility: layer.getVisible()
10   } as LayerManagerEntry;
11 });
12
13 setLayerVisibility(event: any, layerIdentifier: LayerIdentifier): void {
14   if (event.checked) {
15     // show layer
16     this.baseMapService.getLayers().getArray().find(layer => ↩
           layer.get('name') === layerIdentifier).setVisible(true);
17   } else {
18     // hide layer
19     this.baseMapService.getLayers().getArray().find(layer => ↩
           layer.get('name') === layerIdentifier).setVisible(false);
20   }
21 }
```

Vervollständigt wird das Werkzeug-Set durch die Interaktion eine neue Filiale auf der Karte zu setzen. OpenLayers bietet hierzu eine Draw-Interaktion, die es erlaubt einer Layersource ein neues Feature hinzuzufügen.

Codeauszug 4.12: Draw-Interaktion

```
1
2  private addDrawInteraction(): void {
3    this.drawInteraction = new Draw({
4      type: GeometryType.POINT,
5      source: this.tempFilialeLayer.getSource(),
6      style: NEW_STORE_INTERACTION_STYLE
7    });
8
9    this.drawInteraction.on('drawend', (drawEndEvent) => {
10     const feature = drawEndEvent.feature;
11
12     // new feature needs an new ID or OL Fails with ↩
           https://openlayers.org/en/v6.3.1/doc/errors/#30
13     feature.setId(new Date().getTime());
14     featureService.addFiliale(feature);
15     this.deactivateAddFiliale();
16   });
17   this.baseMapService.addInteraction(this.drawInteraction);
18 }
```

In der Initialisierung der Interaktion wird festgelegt, dass nur Punkte erstellt werden können, sowie über einen *NEW_STORE_INTERACTION_STYLE* wird dem Punkt ein eigener Style zugewiesen (ersichtlich in Abbildung 4.1). Ebenso wird ein temporärer Layer für die neu gesetzte Filiale für die Dauer der Interaktion definiert. Da auf der neuen Filiale die Attribute Parkplätze und Verkaufsfläche, sowie die Attraktivität noch nicht gesetzt wurden, ist es einfacher für die Interaktionsdauer einen temporären Layer zu verwenden, der nach Interaktionsende das erstellte Feature an den Feature-Service übergibt, wo die fehlenden Attribute gesetzt werden und das Feature in den Filiallayer übertragen wird. Sobald die Interaktion der Karte hinzugefügt wurde, muss lediglich bei Aktivierung der temporäre Layer der Karte hinzugefügt werden, damit die Interaktion sichtbar ist.

Ebenso müssen andere Interaktionen, vor Allem die Feature-Selektierung der Filialen und Gebiete (Implementierung erfolgt im späteren Verlauf des Kapitels), deaktiviert werden, damit es hierbei nicht zu ungewollten, parallelen Interaktionsaufrufen kommt. Sobald eine neue Filiale gesetzt wurde, muss der Layer wieder entfernt, die Interaktion deaktiviert und andere Interaktionen reaktiviert werden. Aktiviert wird die Interaktion über den entsprechenden Knopf auf der Karte (Icon: *add_icon*, siehe Abbildung 4.1).

Für die Platzierung im DOM werden beide Werkzeuge in einer Werkzeug-Set Komponente zusammengefasst, die ein einheitliches Styling erlaubt.

Als nächstes muss das Objektfenster, welches bei Selektion eines Features geöffnet werden soll und wichtige Informationen über die Filiale oder das Gebiet enthält, implementiert und in den DOM gesetzt werden.
Das Objektfenster ist ein Fenster welches sich wie schon der Layermanager über die Karte legt und Informationen über ein ausgewähltes Feature bietet. Selektiert können sowohl Filialen als auch Zensusgebiete werden. Im Falle einer Filiale soll das Objektfenster auch die Berechnung des Huff-Modells für diese Filiale ermöglichen.
Über einen Objektfenster-Service wird das momentan ausgewählte Feature in Form eines *Obervables*[17] bereitgestellt und ist jederzeit abrufbar. In Angular erfolgt dieser Aufruf über eine gesetzte *Subscription*[18] auf ein Observable. In einer Angular Anwendung werden Komponenten über mehrere Lifecycle-Zyklen instanziiert, die eine besondere Datenübertragung innerhalb der Anwendung erfordern. So erfolgt ein initiales Rendering des Templates und der Daten und Änderungen werden über eine *ChangeDetection* erkannt und neu gerendert. Um Fehler während des ChangeDetection-Lifecycles zu verhindern, erfolgt die Datenübertragung mit Obersvables[19].
Im Fall des Objektfensters subscribed die Komponente sich auf das *currentlySelectedFeature$* des Objektfenster-Services. Dies bedeutet die Komponente erhält über ein Event direkt die neuesten Änderungen sobald sich das Observable ändert. Es erfolgt also kein expliziter, synchroner Datenaufruf, sondern das Observable sendet asynchron Änderungen an alle Subscriber. Somit kann die Selektierung eines neuen Features problemlos an das Objektfenster weitergeleitet werden und die neuen Daten werden korrekt dargestellt.
Das Objektfenster für Filialen zeigt die folgenden Informationen:

[17] *RxJS - Observable* 2021.
[18] *RxJS - Subscription* 2021.
[19] *Angular - Hooking into the component lifecycle* 2021.

- Id
- Parkplätze
- Verkaufsfläche in Quadratmetern
- Marktanteil in Euro [20]
- Marktanteil in Prozent [20]

Für ein Gebiet werden folgende Informationen angezeigt:

- Id
- Einwohner
- Kaufkraft in Euro (Durchschnitt pro Monat pro Person)
- Ausgaben für Lebensmittel (Durchschnitt pro Monat pro Haushalt)
- Wahrscheinlichkeitsfaktor (für den Besuch einer Filiale) [20]
- Marktanteil der Filiale im Gebiet in Euro [20]
- Marktanteil der Filiale im Gebiet in Prozent [20]

Die Informationen zu Parkplätzen und Verkaufsfläche werden über ein Input-Feld dargestellt (siehe Abbildung 4.2). Diese Parameter werden für die Attraktivitätsberechnung der Filiale verwendet und müssen für neue Filialen eingegeben werden. Bei bestehenden Filialen lässt sich somit aber auch die Berechnung des Huff-Modells für die gleiche Filiale mit verschiedenen Parametern ermöglichen.

[20]Nach Berechnung des Huff-Modells

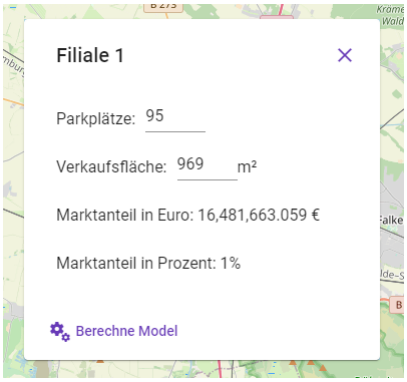

Abbildung 4.2: Bildschirmausschnitt des Objektfensters einer Filiale

Das Objektfenster öffnet sich automatisch nach dem Anlegen einer neuen Filiale. Hierbei werden die Parkplätze und Verkaufsfläche zunächst mit jeweils dem Wert Null angelegt und angezeigt. Bevor das Huff-Modell berechnet werden kann, müssen valide Werte in beide Felder eingetragen werden, da solange der Knopf deaktiviert ist. Die Aktivierung des Knopfes erfolgt über die einfache Validierung der Werte auf eine positive Zahl.
Der Knopf zur Berechnung des Huff-Modells ruft eine Funktion im Feature-Service auf, die die Berechnung für die ausgewählte Filiale durchführen soll. Über den Schließ-Knopf (Icon: *close*, siehe Abbildung 4.2) wird das Objektfenster geschlossen und der Wert des Observables *currentlySelectedFeature$* auf *null* gesetzt.

Mit der Implementierung des Objektfensters sind nun sämtliche Komponenten der Anwendung bereit, die für Berechnung des Huff-Modells und anschließenden farbliche Einfärbung der Zensusgebiete erforderlich sind.

Die Berechnung des Huff-Modells erfolgt im Feature Service. Die für das Huff-Modell erforderliche Parameter Attraktivität A und Faktor der Attraktivität α, sowie Distanz D und Faktor der Distanzfunktion β müssen bestimmt werden. Wie bereits im Kapitel 3.1.1 zum Gravitationsmodell nach Huff erläutert, sind die Faktoren α und β für die Berechnung im realen Wirtschaftsumfeld empirisch anhand von realen Markt- und Wirtschaftsdaten aus Umfragen mit Kunden zu analysieren und zu bestimmen. Für den Prototypen sind diese Werte auf 1,5 für α und 1,5 für β festgelegt. Angenommen wird also, dass je attraktiver eine Filiale ist desto Wahrscheinlicher auch ein Besuch eines Kunden ist. Ebenso, je näher ein Kunde an einer Filiale wohnt desto wahrscheinlicher ist auch hier der Besuch der Filiale.

Die Attraktivität A der Filiale wird über die Summe der Parkplätze und Verkaufsfläche berechnet. Je größer eine Filiale desto größer ist wahrscheinlich das Sortiment und die Menge der Lebensmittel. Und je mehr Parkplätze vor der Filiale vorhanden sind desto mehr Kunden können angezogen werden, die einen weiteren Anreiseweg haben oder mehr auf einmal einkaufen wollen. Für die Berechnung der Attraktivität sind viele weitere Parameter denkbar jedoch bieten die Anzahl der Parkplätze und die Verkaufsfläche in Quadratmetern aus den aufgeführten Gründen eine fundierte Grundlage für die Bewertung der Attraktivität.
Die Distanz D der Filiale zu den Gebieten ist eine einfache Distanzberechnung in OpenLayers vom Standort der Filiale zum Mittelpunkt des jeweiligen Gebiets. Zunächst werden aus den Features der Gebiete über *Type Assertion* Multipolygone über die OpenLayers interne Funktionen zur Mittelpunktbestimmung bieten. Da Multipolygone mehrere Mittelpunkte haben, wird der erste Eintrag aus dem Array gewählt.
OpenLayers bietet für die Längen- und Größenberechnung verschiedene Funktionen, die sich in der Berechnung unter Einbezug der Projektion unterscheiden[21]. Die Koordinaten der Filiale und des Mittelpunkts des Gebiets werden zu einem *LineString* Objekt zusammengefasst[22]. Von dieser Linie wird nun die Länge berechnet. Wird die einfache Berechnung *LineString.getLength()* verwendet, berechnet OpenLayers die Länge anhand von Orthodromen[23] oder der Großkreisdistanz[24]. Wenn auch hervorragend für Flugdistanzen verwendbar, lässt diese Berechnung die Ergebnisse für Entfernungen auf dem Boden ungenau werden.
Über das *Sphere* Paket von OpenLayers lässt sich aber eine Länge anhand der Projektion der Karte berechnen, welches zu den gewünschten Ergebnissen führt[25].

Codeauszug 4.13: Huff-Modell Variablen und Parameter

```
1
2   // distance decay
3   const DIST_DECAY = 1.5;
4   // attractiveness enhancement factor
5   const ATT_ENHANCE_FACTOR = 1.5;
```

[21] *Measure example* 2021.
[22] *OpenLayers v6.5.0 API - Class: LineString* 2021.
[23] *Orthodrome (Großkreis) und Loxodrome* 2021.
[24] *The Great Circle Distance | The Geography of Transport Systems* 2021.
[25] *OpenLayers v6.5.0 API - Module: ol/sphere* 2021.

```
6
7   const centerCoordinates = (gebietFeature.getGeometry() as ↩
        MultiPolygon).getInteriorPoints().getFirstCoordinate();
8
9   const distance = ol_sphere.getLength(new LineString([filialCoordinates, ↩
        centerCoordinates]), {projection: 'EPSG:3857'});
10
11  const attractiveness = filiale.parkingSpaces + filiale.salesArea;
```

Über den Knopf *Berechne Huff-Modell* (Icon: *miscellaneous_services*, siehe Abbildung 4.2) im Objektfenster der ausgewählten Filiale wird die Id der Filiale an den Feature Service übergeben und die Berechnung gestartet. Über die Id wird das Feature der entsprechenden Filiale im Filiallayer identifiziert und kann bearbeitet werden.
Die Wahrscheinlichkeit des Besuchs eines Gebiets für die ausgewählte Filiale innerhalb des gesamten Filialnetzes ergibt sich, wenn die Wahrscheinlichkeit der ausgewählten Filiale ins Verhältnis zur Netzwahrscheinlichkeit gesetzt wird. Für die Filiale errechnet sich die Wahrscheinlichkeit des Besuchs der einzelnen Gebiete daher anhand von zwei *ForEach* Schleifen. Zunächst wird über alle Gebiete iteriert, um die Wahrscheinlichkeit sämtlicher Gebiete für die ausgewählte Filiale zu berechnen. In der Schleife der Gebiete muss für jedes Gebiet über die Filialen iteriert werden, um die Menge der Wahrscheinlichkeiten aller Filialen zu dem Gebiet zu berechnen.

Codeauszug 4.14: Berechnung des Huff-Modells

```
1
2   zensusgebiete.forEach(gebiet => {
3     // probability for all stores
4     let netProbability = 0;
5
6     // probability for the selected store alone
7     const storeProbability = Math.pow(filiale.attractiveness, ↩
          this.ATT_ENHANCE_FACTOR) / Math.pow(distance, this.DIST_DECAY);
8
9     filialen.forEach(store => {
10      netProbability += Math.pow(store.attractiveness, ↩
            this.ATT_ENHANCE_FACTOR) / ↩
```

```
        Math.pow(this.calculateDistancesForFiliale(store.coordinates, ↩
        gebiet.coordinates), this.DIST_DECAY);
    });

    // probability inside net of stores
    gebiet.probability = storeProbability / netProbability;

    // marketshare
    gebiet.marketShare = gebiet.spendituteGroc * gebiet.probability;
    gebiet.marketSharePercentage = gebiet.marketShare / ↩
        gebiet.spendituteGroc;
  });
```

Nachdem jedes Gebiet Besuchswahrscheinlichkeiten ausgerechnet bekommen hat, wird der Marktanteil prozentual sowie monetär für ein besseres Verständnis sowie eine bessere Einordnung errechnet. Hierzu werden die Ausgaben des Gebiets für Lebensmittel im Monat mit der Wahrscheinlichkeit des Besuchs multipliziert, um den Anteil der Lebensmittelausgaben in Euro des Gebiets für die ausgewählte Filiale zu errechnen. Wird dieser Betrag nun ins Verhältnis zu den Gesamtausgaben für Lebensmittel im Gebiet gesetzt, errechnet sich der prozentuale Anteil. Erfolgten diese Schritte für alle Gebiete und werden die Beträge des Anteils der Ausgaben der Filiale und der Gesamtausgaben pro Gebiet summiert, ergibt sich der Marktanteil der Filiale im Netz in Euro und, im Verhältnis zu einander, in Prozent.

Codeauszug 4.15: Martkanteil Berechnung

```

  let marketShareTotal = 0;
  let totalMarketExpenditure = 0;
  this._zensusMap.forEach((gebiet: ZensusProperties) => {
    marketShareTotal += gebiet.marketShare;
    totalMarketExpenditure += gebiet.spendituteGroc;
  });
  filiale.marketShare = marketShareTotal;
  filiale.marketSharePercentage = marketShareTotal / totalMarketExpenditure;
```

Bei jeder Berechnung der Wahrscheinlichkeit des Besuchs der Gebiete für die ausgewählte Filiale wird final nun noch das Gebiet eingefärbt. Hierzu wird auf jedem Feature abhängig der errechneten Wahrscheinlichkeit ein Indikator gesetzt, der einer bestimmten Farbe entspricht.

Codeauszug 4.16: Setzen des Einfärbeindikators

```
1
2   const probabilityInPercent = gebiet.probability * 100;
3
4   if (probabilityInPercent > 80) {
5     gebiet.indicator = 1;
6   }
7   else if (probabilityInPercent < 80 && probabilityInPercent > 70) {
8     gebiet.indicator = 2;
9   }
10
11  ...
12
13  else {
14    gebiet.indicator = 9;
15  }
```

Dieser Indikator wird im Renderzyklus der Gebiete innerhalb der *StyleFunction* pro Gebiet ausgelesen und das Gebiet entsprechend eingefärbt (ähnlich zur StyleFunction der Filialen 4.9).

Codeauszug 4.17: Einfärben der Gebiete

```
1
2   protected readonly colorGradient: ColorInterface[] = [
3     { gravitationalRing: 1, value: 'rgb(255, 0, 0)'},
4     { gravitationalRing: 2, value: 'rgb(255, 70, 0)'},
5     { gravitationalRing: 3, value: 'rgb(255, 105, 0)'},
6     { gravitationalRing: 4, value: 'rgb(255, 134, 0)'},
7     { gravitationalRing: 5, value: 'rgb(255, 160, 0)'},
8     { gravitationalRing: 6, value: 'rgb(246, 185, 0)'},
9     { gravitationalRing: 7, value: 'rgb(241, 206, 31)'},
```

```
10      { gravitationalRing: 8, value: 'rgb(236, 227, 66)'},
11      { gravitationalRing: 9, value: 'rgb(231, 247, 98)'}
12    ];
13
14    const gebieteStyle = new Style({
15      stroke: new Stroke({
16        color: 'white',
17        width: 2,
18        lineDash: [0.1, 7]
19      }),
20      fill: new Fill({
21        color: 'rgba(52, 164, 235, 0.5)'
22      })
23    });
24
25    const gebieteStyleFunction = (feature, resolution) => {
26      const indicator = feature.indicator;
27
28      if (!indicator) {
29        return this.ezbLineStyle;
30      }
31      const color = colorGradient.find(color => color.gravitationalRing === ↩
            indicator).value;
32      gebieteStyle.getFill().setColor(color);
33      return gebieteStyle;
34    }
```

Für die Farben der einzelnen Gravitationsringe wurde ein Farbgradient von Rot zu Gelb gewählt. Besonders hohe Besuchswahrscheinlichkeiten werden mit einer Farbe aus dem roten Spektrum des Gradienten dargestellt und mit abnehmender Wahrscheinlichkeit wird das Gebiet zunehmend gelber. Das berechnete Modell kann in der Abbildung 4.3 begutachtet werden.

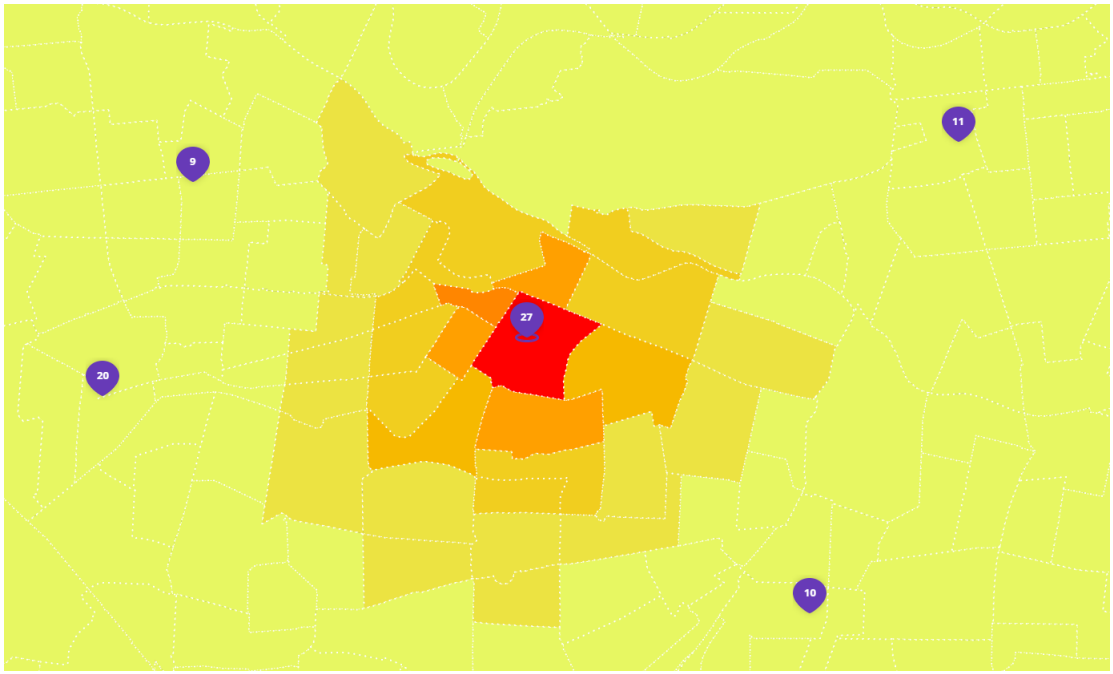

Abbildung 4.3: Bildschirmausschnitt des Prototypen nach der Berechnung des Huff-Modells für Filiale 27

Kapitel 5

Auswertung

Im letzten Teil dieser Arbeit wird das Huff-Modell und dessen Umsetzung im Prototypen ausgewertet.
Was sind die Anwendungsfälle des Huff-Modells? Wozu kann der Prototyp verwendet werden? Ist der Prototyp performant? Ist das Huff-Modell eine gute Wahl für einen Algorithmus der Filialplanung und Standortanalyse? Wo stößt das Modell an seine Grenzen? Was kann verbessert, erweitert oder ersetzt werden?

Unter Anderem diesen Fragen wird mit Ausblick auf die Zukunft kritisch Antwort gegeben bevor ein finales Fazit gezogen wird.

5.1 Bewertung

Bei der Bewertung des Prototypen muss einerseits die Umsetzung des Huff-Modells in einer Webkarte mit modernen Technologien und andererseits das Modell selber bewertet werden. Hierzu werden zunächst Anwendungsfälle des Modells beschrieben, die Verwendung des Prototypen unter Einbeziehung der Performance für diese erklärt und Verbesserungen und Erweiterungen vorgestellt. Danach wird das Modell selber in Frage gestellt und mögliche Alternativen oder Erweiterungen vorgestellt.

Obwohl das Huff-Modell kein neues Phänomen ist und schon mehrere Jahrzehnte alt ist, hat es sich als verlässlich erwiesen und findet nach wie vor Gebrauch. Die liegt vor Allem an der einfachen und schnellen Verwendung. Bereits mit wenig Informationen können

simple Prognosen zu Besucherwahrscheinlichkeit erstellt werden.
Zu den Hauptanwendungsfällen zählen:

- Filialplanung
- Expansionsplanung
- Verteilgebietsplanung
- Kundenansprache

Die Filialplanung und Expansionsplanung sowie die Verteilgebietsplanung und die Kundenansprache sind ähnlich zu betrachten, da sie sich nur geringfügig Unterscheiden.
Der Prototyp kann alle aufgelisteten Fälle grundlegend abdecken,könnte aber aber bei gezielten Erweiterungen eine noch bessere Unterstützung für die spezifischen Anforderungen bieten. So lassen sich Besuchswahrscheinlichkeiten in bestehendem Netz sowie mit neuen Filialen berechnen und vergleichen. Potenzielle neue Standorte können dem bestehenden Netz hinzugefügt werden und deren Auswirkung auf Wahrscheinlichkeiten und Umsatz werden direkt visuell und numerisch sichtbar, was bei direktem Vergleich eine Erstellung einer Reihenfolge anhand von Umsatzgewinn oder -verlust und Marktanteil der potenziellen neuen Standorte erlaubt.
Hierzu wäre eine Erweiterung hinsichtlich direkter Gegenüberstellung der potenziellen Standorte in Form einer Tabelle oder Ähnlichem denkbar. Im Prototyp lassen sich zwar beliebig viele neue Filialen setzen, jedoch gibt es keine Möglichkeit die gesetzten Filialen und deren Einfluss auf Wahrscheinlichkeiten und Marktanteil zu vergleichen.

Ebenso kann sehr leicht ein Verteilgebiet für Handzettel (Werbung) abgegrenzt werden, dass sich auch wirklich nur an Kunden richtet, die wahrscheinlich in der Filiale einlaufen würden. Außerdem sind aus den Gebieten des Verteilgebiets erste Informationen zu Kaufkraft, Einwohneranzahl und Ausgaben ersichtlich, welches eine persönlichere Ansprache der Kunden zulässt, die auf demografischen Analysen beruht.
Eine sinnvolle Erweiterung für die Verteilgebietserstellung wäre die Gruppierung mehrerer Gebiete zu einem neuen Verteilgebiet in einem neuen Layer. Das neue Gebiet könnte dann eine eigene Farbe erhalten und nach Bedarf ein- und ausgeblendet werden oder in der Z-Ebene nach oben oder unten verschoben werden. Für das neue Gebiet können dann die

Daten der eingeschlossenen Gebiete summiert werden und für das potenzielle Verteilgebiet angezeigt werden.

Die Berechnung des Modells ist bereits angemessen performant mit initialer Lade- und Renderzeit von unter 1 Sekunde für die Gebiete und Filialen sowie Berechnungs- und Rerenderzeit der Gebiete von ebenfalls unter 1 Sekunde. Durch Optimierung beim Laden und Rendern durch zum Beispiel explizite Nutzung von Cache-Objekten für Features und Styles oder Überarbeitung des Codes für die Berechnung des Modells, könnte hier eventuell noch ein wenig mehr Performance optimiert werden.

Der Prototyp kann das Huff-Modell sehr gut darstellen und bietet echten Mehrwert in Prozessen des Geomarketings. Dennoch hat das Modell auch einige Schwächen oder Ungenauigkeiten, wie bereits im Artikel Äpplication of network Huff model for commercial network planning at suburban – Taking Wujin district, Changzhou as a case"von Haozhi Pan,Yongfu Li und Anrong Dang beschrieben[1]. Die Autoren stellen vor Allem die Bestimmung der Parameter α und β sowie die Berechnung der Distanzvariable D als problematisch hervor. Während es für die Bestimmung der Parameter meist an regionalen Marktdaten mangelt oder eine Regressionsanalyse mit Linearisierung nur schwer durchzuführen ist, so gibt es schlichtweg mehrere Möglichkeiten die Distanz zwischen Filiale und Kunde zu berechnen. Die Distanz kann euklidisch dargestellt werden oder über die Netzwerkdistanz oder - zeit (Fahrzeitzone in Meter oder Minuten) wobei alle drei zu verschiedenen Ergebnissen des Modell führen. Ebenso finden sich direkt im Prototypen einige Schwächen wieder.
So lassen sich Filialen finden, bei denen die Berechnung zu einem ungenauen Ergebnis, wie in Abbildung 5.1 zu sehen, führt. In der Abbildung ist klar zu erkennen, dass das Gebiet in welchem sich die Filiale befindet selbst nicht die höchste Besuchswahrscheinlichkeit hat. Diese Ungenauigkeit lässt sich auf die Berechnung des Mittelpunkts des Gebiets zurückführen, so ist die Filiale näher am Mittelpunkt des Nachbargebiets als am Mittelpunkt des eigenen Gebiets gelegen, was zu einer höheren Besuchswahrscheinlichkeit des Nachbargebiets führt.
Dieses Problem lässt sich relativ einfach mit der Verwendung von kleineren Gebieten eindämmen. Jedoch ist die ideale Lösung mit der Verwendung von Punkten pro Haushalten nicht wirklich umsetzbar, da die Datenmenge um ein Vielfaches größer wäre und die

[1] Pan, Li und Dang 2013.

Berechnung zu viel Ressourcen benötigen oder zu lange dauern würde.

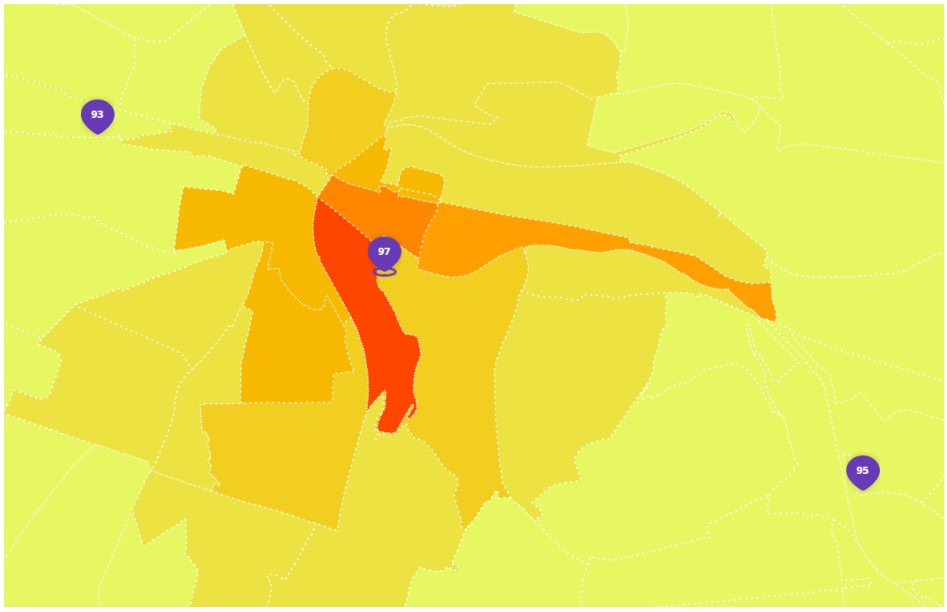

Abbildung 5.1: Bildschirmausschnitt des Prototypen mit ungenauer Berechnung

Weiterführend können zwar die Attraktivität und Distanz nahezu beliebig genau Errechnet werden, jedoch werden auch nur genau diesen beiden Variablen in die Berechnung einbezogen. Aufgrund dieser Einschränkung und vor Allem den bereits aufgeführten Problemen in Bezug auf die Bestimmung der Parameter α und β war das Huff-Modell im Laufe der Jahre bereits mehrfach Kern wissenschaftlicher Arbeiten, die sich mit der Validierung und Verbesserung des Modells beschäftigten.
So entstand 1974 das Multiplicative Competitive Interaction Modell (kurz MCI-Modell), welches genau diese Probleme behandelt[2].

5.2 Fazit/ Ausblick

Das Huff-Modell als Grundlage der algorithmischen Berechnung in dem Prozess einer Filialplanung erweist sich als einfach zu verstehendes, schnelles und nützliches Werkzeug, um mit relativ wenig Marktdaten bereits eindeutige und aussagekräftige Informationen

[2]Nakanishi und Cooper 1974.

über mögliche neue Standorte einer Filiale zu erhalten.
Im Prototypen wurden die verfügbaren Marktdaten um lediglich ein paar wenige Attribute ergänzt, um eine realistische Abbildung eines Filialnetzes innerhalb Berlins zu generieren. Die Parameter der Berechnung lassen sich hierbei einfach fast beliebig Erweitern, so können in die Berechnung der Attraktivität deutlich mehr Attribute einer Filiale oder des Umfelds einfließen. Denkbar wären zum Beispiel die Ergänzung von Informationen über umliegende Sehenswürdigkeiten (engl. Point of interest), die Eingliederung der Filiale in ein Einkaufszentrum, Markenstärke einer Filialkette oder tatsächliches Sortiment, welche alle Einfluss auf die Attraktivität einer Filiale nehmen.
Ebenso kann die Berechnung der Distanz auf komplexeren Grundlagen wie zum Beispiel der Fahrzeit mit dem Auto, den öffentlichen Verkehrsmitteln oder dem Fahrrad basieren. Bereits im Kapitel3.1.1 Gravitationsmodell erwähnt, sollten die Parameter α und β empirisch bestimmt werden. Der Komplexität der Erhebung der Parameter ist hierbei keine Grenze gesetzt. Kundenbefragungen, amtliche Statistiken oder Meinungsbilder aus der freien Wirtschaft können hier als Grundlage dienen.
Für die Anwendung in echten Szenarien der freien Wirtschaft sollten also unbedingt eine verfeinerte Kalibrierung der Berechnung anhand eigener Daten erfolgen, damit der Prototyp zum vollen Potenzial verwendet wird.

Als Komponente eines größeren, komplexeren GIS, würde der Prototyp einzigartige, ergänzende Features bieten, die die Prozesse innerhalb der Standortanalyse und Filialplanung deutlich vereinfachen.

Abbildungsverzeichnis

[3] *Transform coordinates - GPS online convertor* 2021.

Quelltextverzeichnis

Literaturverzeichnis

Angular - Hooking into the component lifecycle (2021). URL: https://angular.io/guide/lifecycle-hooks (besucht am 19. 02. 2021).

Angular CLI (2021). URL: https://cli.angular.io/ (besucht am 12. 02. 2021).

Angular Material (2021). en-US. URL: https://material.angular.io/ (besucht am 19. 02. 2021).

Angular Material Button (2021). en-US. URL: https://material.angular.io/components/button/overview (besucht am 19. 02. 2021).

Angular Material Card (2021). en-US. URL: https://material.angular.io/components/card/overview (besucht am 19. 02. 2021).

Angular Material Checkbox (2021). en-US. URL: https://material.angular.io/components/checkbox/overview (besucht am 19. 02. 2021).

Angular Material Icon (2021). en-US. URL: https://material.angular.io/components/icon/overview (besucht am 19. 02. 2021).

Angular Material Input (2021). en-US. URL: https://material.angular.io/components/input/overview (besucht am 19. 02. 2021).

Angular Material List (2021). en-US. URL: https://material.angular.io/components/list/overview (besucht am 19. 02. 2021).

Anlage zu Nr. 51.11 VV BauO NRW Richtzahlen für den Stellplatzbedarf (2021). de. URL: https://recht.nrw.de/lmi/owa/br_vbl_show_pdf?p_id=528 (besucht am 29. 01. 2021).

Definition Geomarketing | Geomarketing (2021). de-DE. URL: https://www.geomarketing.de/was-ist-geomarketing/ (besucht am 11. 02. 2021).

Geodaten | Geomarketing (2021). de-DE. URL: https://www.geomarketing.de/geodaten/ (besucht am 11. 02. 2021).

Geographische Informationssysteme (GIS) | Geomarketing (2021). de-DE. URL: https://www.geomarketing.de/geomarketing-software/geografische-informationssysteme/ (besucht am 12. 02. 2021).

Geoportal / Land Berlin (2021). URL: https://www.stadtentwicklung.berlin.de/geoinformation/ (besucht am 15. 02. 2021).

GfK Kaufkraft je Einwohner nach Bundesländern 2021 (2021). de. URL: https://de.statista.com/statistik/daten/studie/168591/umfrage/kaufkraft-nach-bundeslaendern/ (besucht am 22. 01. 2021).

GIS-Standards (2021). de. URL: https://www.gistandards.eu/de/gis-standards/ (besucht am 11. 02. 2021).

Haas, Prof Dr Hans-Dieter (2021). *Definition: Standortfaktoren*. de. Publisher: Springer Fachmedien Wiesbaden GmbH Section: economy. URL: https://wirtschaftslexikon.gabler.de/definition/standortfaktoren-45787/version-133523 (besucht am 11. 02. 2021).

Herter, Michael und Karl-Heinz Mühlbauer, Hrsg. (2018). *Handbuch Geomarketing*. ger. 2. Aufl. Wichmann Verlag. ISBN: 978-3-87907-654-3. URL: https://content-select.com/de/portal/media/view/5c7e7232-1828-4831-bb3c-7986b0dd2d03.

ISO 19115 (2021). *ISO 19115:2003*. en. URL: https://www.iso.org/cms/render/live/en/sites/isoorg/contents/data/standard/02/60/26020.html (besucht am 11. 02. 2021).

Jung, Karl (1956). „Figur der Erde“. In: *Geophysik I / Geophysics I*. Hrsg. von J. Bartels. Berlin, Heidelberg: Springer Berlin Heidelberg, S. 534–639. ISBN: 978-3-642-45855-2. DOI: 10.1007/978-3-642-45855-2_16. URL: https://doi.org/10.1007/978-3-642-45855-2_16.

Konsumausgaben in Deutschland für Nahrungsmittel, Getränke und Tabakwaren bis 2019 (2020). de. Library Catalog: de.statista.com. URL: https://de.statista.com/statistik/daten/studie/161565/umfrage/konsumausgaben-der-privaten-haushalte-in-deutschland-fuer-nahrungsmittel-zeitreihe/ (besucht am 03. 06. 2020).

Konsumausgaben privater Haushalte in Deutschland (2021). de. URL: https://www.destatis.de/DE/Themen/Gesellschaft-Umwelt/Einkommen-Konsum-Lebensbedingungen/Konsumausgaben-Lebenshaltungskosten/Tabellen/privater-konsum-d-lwr.html (besucht am 22. 01. 2021).

Lebensmittel-Discounter in Deutschland bis 2018 (2020). de. Library Catalog: de.statista.com. URL: https://de.statista.com/statistik/daten/studie/459711/umfrage/lebensmittel-discounter-in-deutschland/ (besucht am 03. 06. 2020).

Lidl - Verkaufsfläche der Discountmärkte in Deutschland | Zeitreihe | Handelsdaten.de | Statistik-Portal zum Handel (2021). URL: https://www.handelsdaten.de/lebensmitteldiscounter/durchschnittliche-verkaufsflaeche-der-discountmaerkte-lidl-deutschland (besucht am 29. 01. 2021).

Marktdaten | Geomarketing (2021). de-DE. URL: https://www.geomarketing.de/marktdaten/ (besucht am 12. 02. 2021).

Measure example (2021). URL: https://openlayers.org/en/latest/examples/measure.html (besucht am 22. 02. 2021).

Nakanishi, Masao und Lee G. Cooper (1974). „Parameter Estimation for a Multiplicative Competitive Interaction Model—Least Squares Approach". In: *Journal of Marketing Research* 11.3. _eprint: https://doi.org/10.1177/002224377401100309, S. 303–311. DOI: 10.1177/002224377401100309. URL: https://doi.org/10.1177/002224377401100309.

OpenLayers v6.5.0 API - Class: LineString (2021). URL: https://openlayers.org/en/latest/apidoc/module-ol_geom_LineString-LineString.html (besucht am 22. 02. 2021).

OpenLayers v6.5.0 API - Class: Map (2021). URL: https://openlayers.org/en/latest/apidoc/module-ol_Map-Map.html (besucht am 12. 02. 2021).

OpenLayers v6.5.0 API - Module: ol/sphere (2021). URL: https://openlayers.org/en/latest/apidoc/module-ol_sphere.html (besucht am 22. 02. 2021).

Origins of the Canada Geographic Information System | ArcNews (2020). URL: https://www.esri.com/news/arcnews/fall12articles/origins-of-the-canada-geographic-information-system.html (besucht am 21. 09. 2020).

Orthodrome (Großkreis) und Loxodrome (2021). de-DE. URL: https://www.frassek.org/3d-mathe/orthodrome-gro%C3%9Fkreis-und-loxodrome/ (besucht am 22. 02. 2021).

Pan, Haozhi, Yongfu Li und Anrong Dang (Sep. 2013). „Application of network Huff model for commercial network planning at suburban – Taking Wujin district, Changzhou as a case". In: *Annals of GIS* 19.3. Publisher: Taylor & Francis _eprint: https://doi.org/10.1080/19475683.2013.806356, S. 131–141. ISSN: 1947-5683. DOI: 10.1080/19475683.2013.806356. URL: https://doi.org/10.1080/19475683.2013.806356 (besucht am 26. 02. 2021).

RxJS - Observable (2021). en. URL: https://rxjs-dev.firebaseapp.com/guide/observable (besucht am 19. 02. 2021).

RxJS - Subscription (2021). en. URL: https://rxjs-dev.firebaseapp.com/guide/subscription (besucht am 19. 02. 2021).

Standortanalysen | Geomarketing (2021). de-DE. URL: https://www.geomarketing.de/geomarketing-analysen-a-z/standortanalyse/ (besucht am 11. 02. 2021).

The Great Circle Distance | The Geography of Transport Systems (2021). en-US. URL: https://transportgeography.org/contents/chapter1/transportation-and-space/great-circle-distance/ (besucht am 22. 02. 2021).

Transform coordinates - GPS online convertor (2021). en. URL: http://epsg.io (besucht am 12. 02. 2021).

Untitled Document (Okt. 2005). URL: https://web.archive.org/web/20051030072118/http://www.gis.dce.harvard.edu/fisher/HTFisher.htm (besucht am 21. 09. 2020).

Verkehrszellen - Berlin (2021). de-de. URL: https://hub.arcgis.com/datasets/esri-de-content::verkehrszellen-berlin?geometry=13.098,52.471,13.751,52.544 (besucht am 15. 02. 2021).

WGS 84 - WGS84 - World Geodetic System 1984, used in GPS - EPSG (2021). *WGS 84 - WGS84 - World Geodetic System 1984, used in GPS - EPSG:4326*. en. URL: http://epsg.io (besucht am 12. 02. 2021).

WGS 84 / Pseudo-Mercator - Spherical Mercator, Google Maps, OpenStreetMap, Bing, ArcGIS, ESRI - EPSG (2021). *WGS 84 / Pseudo-Mercator - Spherical Mercator, Google Maps, OpenStreetMap, Bing, ArcGIS, ESRI - EPSG:3857*. en. URL: http://epsg.io (besucht am 12. 02. 2021).

Anhang A

A.1 Diagramm

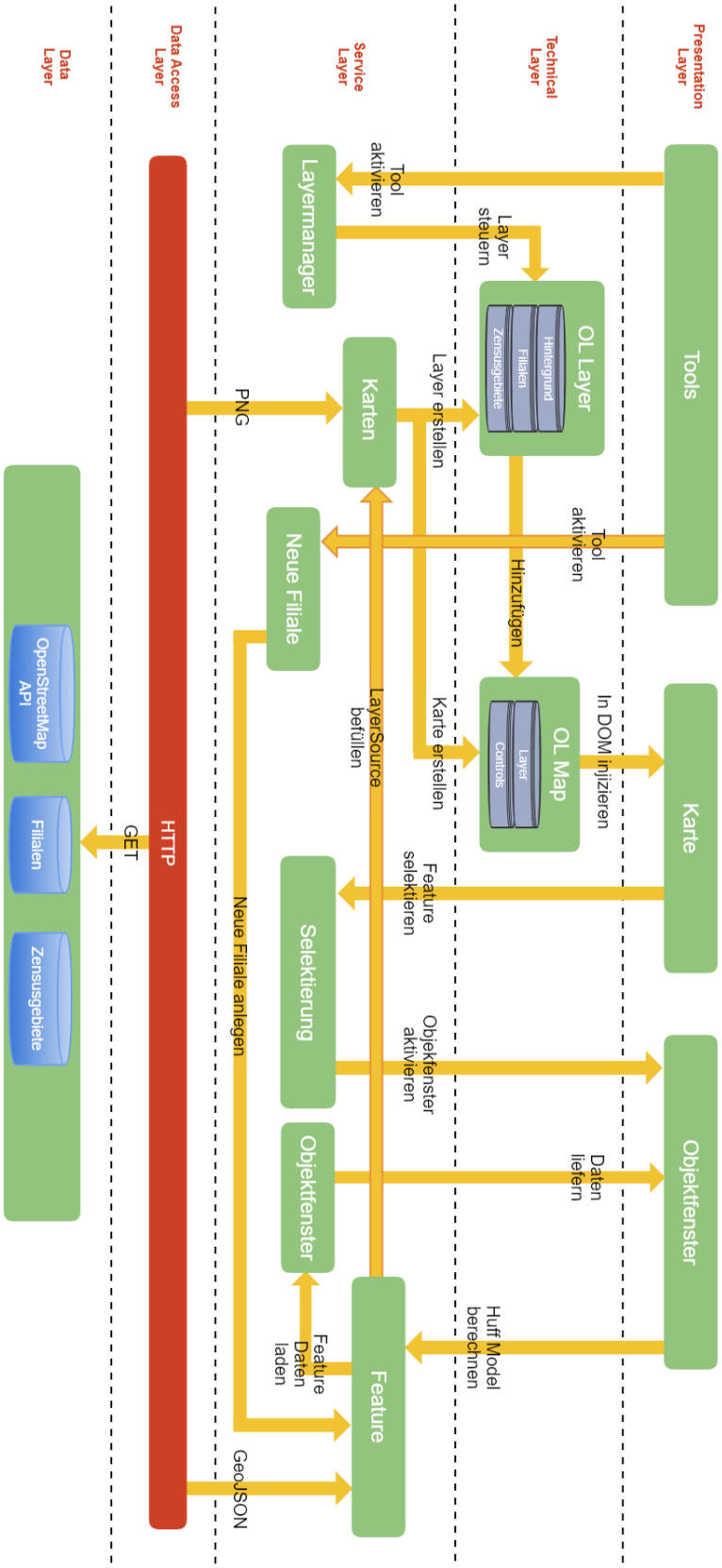

Abbildung A.1: Architekturbild

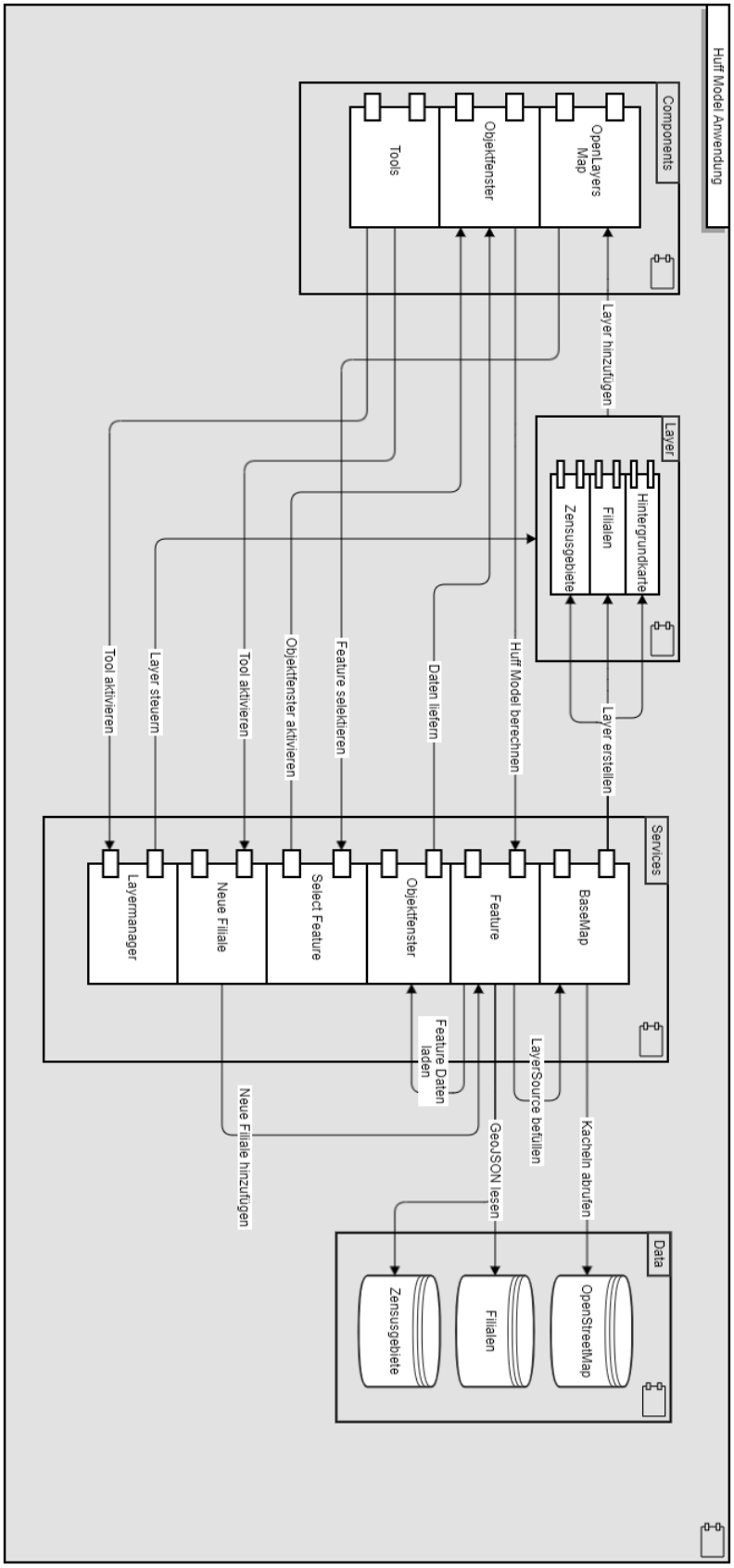

Abbildung A.2: Komponentendiagramm