Institut für Visualisierung und Interaktive Systeme
Universität Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Diplomarbeit Nr. 3148

Visualisierung für webbasierte Kartendienste: Entwurf und Implementierung einer Visualisierungskomponente für Google Maps

Stefan Wokusch

Studiengang:	Softwaretechnik
Prüfer:	Prof. Dr. Thomas Ertl
Betreuer:	MSc. Harald Bosch Dipl. Inf. Dennis Thom
begonnen am:	18. Februar 2011
beendet am:	22. August 2011
CR-Klassifikation:	H.1.2, H.5.2, I.3.8

Zusammenfassung

Plant man eine Aktivität unter freiem Himmel, spielen viele Umwelteinflüsse eine Rolle. Neben Ausflugsplanungen in freier Natur, ist auch die tägliche Fahrt zur Arbeit, die je nach Wetterlage, mit dem Fahrrad oder dem Auto zurückgelegt werden soll, in dieser Arbeit von Interesse.

Um die Umwelteinflüsse übersichtlich und zusammengefasst darzustellen, werden verschiedene Darstellungsmöglichkeiten in der Arbeit diskutiert. Auch spielen Anpassungsmöglichkeiten, an den Benutzer und die von ihm auszuführende Aktivität, eine Rolle. Die gleichzeitige Anzeige mehrerer Darstellungen, ist ebenfalls ein Schwerpunkt.

Als Resultat entsteht ein lauffähiger Prototyp, mit dem man die verschiedenen Kombinationen von Darstellungen ausprobieren und einstellen kann. Dieser ist mit Hilfe des GWT(Google Web Toolkit)-Frameworks implementiert und kann Umwelteinflüsse, wie beispielsweise Wind-, Temperatur- und Luftqualitäts-Daten, auf verschiedene Art und Weise, darstellen. Zudem werden Implementierungsdetails des Systems und einzelner Darstellungen näher erklärt.

Abschließend folgt eine Benutzerstudie, die zunächst die Darstellungen einzeln und am Ende deren Kompatibilität näher untersucht. Dabei werden weiterführende Erkenntnisse gewonnen, wie beispielsweise, dass die zusätzliche Anzeige der exakten Werte, in einer separaten Ansicht, zur besseren einschätzbarkeit der Daten beiträgt.

Abstract

Planning of activities in the open, is influenced by many environmental conditions. Trips in the nature depending on the weather conditions, like excursions in the nature or the daily commute to work, which can be done by bicycle or by car, are of interest in this work.

To display environmental influences, in a clear structured form, different presentation options are evaluated. Adaptation to the user and his activities as well as the dislplay of multiple presentations is another key part.

The proposed techniques were imlemented in a runnable prototype, that is able to show and configure combinations of diffrent views. This prototype is implemented with the GWT(Google Web Toolkit)-Framework and is able to show environmental conditions, like wind-, temperature and airquality, in different ways. In addition, implementation details of the system and single algorithms are explained in more detail.

Finally, a user research study follows where the Views and their compatibility, will be evaluated. Further results of this study are for example that showing the exact values in a seperate View increases the assessability of the data.

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

1.1 Motivation

Plant man eine Aktivität unter freiem Himmel, spielen viele Umwelteinflüsse eine Rolle. Diese muss man heutzutage von vielerlei Diensten zusammentragen und mühsam analysieren. Vor allem wenn man beispielsweise eine mehrstündige Fahrradtour plant, stößt man schnell an die Grenzen heutiger Informationssysteme. Das PESCaDO-System soll darauf eine Lösung sein, indem es die relevanten Daten zusammenträgt und nach dem Profil des Benutzers diese alle auf einen Blick darstellt. In dieser Arbeit werden dabei die verschiedenen Darstellungsmöglichkeiten eines solchen Systems näher beschrieben.

Vor allem die Anpassungen an den Anwender sind sehr interessant. Kennt das System beispielsweise die Gesundheit und Belastbarkeit des Anwenders, kann es die kritischen Daten besser hervorheben oder die Skalen an den Benutzer und die zu planende Aktivität genau anpassen.

1.2 Aufgabenstellung

Ziel dieser Arbeit ist die Entwicklung einer GWT(Google Web Toolkit)-basierten Visualisierungskomponente für das PESCaDO-System. Diese Komponente visualisiert mit Hilfe von GoogleMaps und HTML5 Elementen verschiedene Umwelteinflüsse, um dem Benutzer situationsbedingte Entscheidungen zu ermöglichen. Dabei geht es vor allem um die Darstellung von Wind-, Temperatur- und Luftqualitäts-Daten. In dieser Arbeit wird die gleichzeitige Darstellung möglichst vieler Daten im Vordergrund stehen. Da der Anwender meist die Daten auf einer Route wissen will, muss auch die zeitliche Komponente der Daten berücksichtigt und dargestellt werden. Im System werden hauptsächlich zukünftige Aktivitäten des Anwenders geplant und analysiert. Daher wird auch die Unsicherheit der Daten dargestellt werden, um dem Anwender keine realen Daten zu suggerieren.

Am Ende entsteht ein lauffähiger Prototyp, mit dem man die verschiedenen Kombinationen von Darstellungenausprobieren und einstellen kann. Darauf aufbauend wird eine Benutzerstudie folgen, um die Darstellungen und deren Kompatibilität zu untersuchen.

1.3 Lösungsansatz

Am Anfang wird analysiert werden, was für Daten relevant sind und wie diese in sinnvollen Gruppen zusammengestellt werden können. Dabei wird ebenfalls auf verschiedene Arten von Unsicherheit eingegangen, um dies im Modell berücksichtigen zu können.

Nachdem das Datenmodell fertig beschrieben ist, werden die unterschiedlichen Darstellungsformen und deren Eigenschaften analysiert. Dabei werden auch verschiedene Darstellungen von Unsicherheit berücksichtigt.

Da am Ende der Arbeit ein lauffähiger Prototyp entsteht, werden dann die Anforderungen an das System spezifiziert und die verwendeten Technologien, Architektur und andere Implementierungsdetails dargestellt. Zudem wird ausgeführt, wie man das System später in bestehende Anwendungen integrieren kann.

2 Grundlagen

2.1 Farben

2.1.1 Signalfarben

Farben stellen nicht nur eine Kodierungsmöglichkeit dar, sie vermitteln auch unbewusst noch weitere Informationen. Die Farbe Rot beispielsweise steht für Gefahr und erweckt so die Aufmerksamkeit des Betrachters. Grün dagegen steht unter anderem für Gesundheit, Leben oder OK [Daho6]. Ein gutes Beispiel, bei der diese Signalwirkung der Farben verwendet wird, ist eine Verkehrsampel.

Blau hat ebenfalls eine besondere Bedeutung, die in unserem Kontext wichtig ist. Blau steht für Kälte [Daho6] und sollte somit für niedrige Temperaturwerte verwendet werden.

2.1.2 Rot-Grün-Blindheit und andere Schwächen

Was es bei Farben zu beachten gibt, ist, dass es mehrere Arten von Farbfehlsichtigkeit gibt. Dabei können die betroffenen Menschen einzelne Farben nicht oder nur schwer voneinander unterscheiden. Rot-Grün-Schwäche, auch Protanopie (Rot-Schwäche) bzw. Deuteranopie (Grün-Schwäche) genannt, ist hier besonders zu beachten, da sie sehr verbreitet ist. Sie betrifft ca. 8% der männlichen und ca. 0,5% der weiblichen Bevölkerung [Gego6]. In seltenen Fällen kann es auch vorkommen, dass gar keine Farben mehr unterschieden werden können.

Aus diesem Grund ist es sinnvoll, Farben nur unterstützend einzusetzen. Die Informationen sollten auch anders kodiert dargestellt werden, wie beispielsweise mit der Position bei einer Verkehrsampel.

2.2 Streamline, Streakline und Pathline

Streamline, Streakline und Pathline sind Möglichkeiten um zweidimensionale Vektorfelder, wie beispielsweise **Windrichtungen**, darzustellen. Dazu zeichnet man auf einem gleichmäßigen Raster kurze Linien oder Glyphen und richtet sie an den Vektoren aus. Eine weitere Möglichkeit sind Streamlines. Diese zeigen, wie sich ein Partikel bewegen würde, wenn sich die Windrichtungen nicht ändern würden. Fügt man nun eine Veränderung über die Zeit ein, so müssen die Streamlines entweder animiert werden oder man verwendet alternative Anzeigemöglichkeiten. Eine ist die Streakline. Dies kann als Partikelspur angesehen werden,

bei dem ein Partikel hinter sich weitere Partikel erzeugt, die ebenfalls dem Wind ausgesetzt sind. Setzt man diese Partikel nicht dem Wind aus, entsteht eine Historie des Weges des Partikels, die als Pathline bezeichnet wird [TWHS04].

Man kann mit diesen Darstellungen zwar die Richtung des Windes gut erkennen, aber die **Windgeschwindigkeit** ist schwer abzulesen. Eine Möglichkeit ist die Verwendung von gestrichelten Linien. Indem man die Striche in der Länge oder Stärke verändert, lässt sich die Windgeschwindigkeit daran ablesen. Hat man die Möglichkeit von Animation, so kann man auch die Path- oder Streakline abschneiden und durch ihre Gesamtlänge die Geschwindigkeit darstellen. Dabei haben die Partikel in der Spur bei der Streakline eine Lebenszeit, wodurch sich die Länge der Spur automatisch verkürzt, wenn das vorderste Partikel langsamer wird. Ein Nachteil daran ist, dass die aktuelle Geschwindigkeit des Teilchens nicht ablesbar ist. Stattdessen ist die Durchschnittsgeschwindigkeit seit der Lebenszeit des letzten Partikels gut zu erkennen.

Für unseren Anwendungsfall ist die animierte Darstellung einer Pathline ausreichend. Da man nur einen guten Überblick über die Windverhältnisse erhalten will, stört die resultierende Anzeige der Durchschnittsgeschwindigkeit kaum, solange die Pathlines nicht zu lang sind.

2.3 GWT

GWT ist die Abkürzung für das Google Web Toolkit[1]. Dies enthält, neben einem GUI-Framework, einen Compiler, um Javacode in Javascript umzuwandeln [Muro7]. Damit ist es einfacher möglich, komplexe Systeme in einer Browserumgebung plattformübergreifend laufen zu lassen, ohne dass der Nutzer etwas zusätzlich installieren muss.

2.4 Asyncron/Callback Pattern

Das Callback Pattern[2] ist für die Kommunikation mit anderen Systemen geeignet. Hierbei wird eine Callback-Funktion übergeben, die aufgerufen wird, sobald eine Antwort des angefragten Systems verfügbar ist [GHJV95].

Dies ist besonders für JavaScript relevant, da dies zum Zeitpunkt dieser Arbeit zusammen mit dem Renderer des Browsers in einem einzelnen Thread abläuft. Javascript ist also blockierend und kann keine synchronen Serverabfragen durchführen.

[1] http://code.google.com/intl/de-DE/webtoolkit/
[2] http://de.wikipedia.org/w/index.php?title=Remote_Procedure_Call&stableid=88509639

2.5 Injection Framework

Ein weit verbreitetes Entwurfsmuster ist Dependency Injection[3] [Pra09]. Dies dient dazu, die Abhängigkeiten und Instanzen von Systemen zu verwalten. Es bietet die Möglichkeit, einzelne Komponenten von Frameworks nachträglich zu ersetzen, ohne dass Eingriffe in den Code dieser Klassen notwendig sind.

Das Konzept ist, sämtliche „new"-Operatoren von separaten Klassen im Konfigurationsteil der Software zu machen. In diesen sogenannten Factories kann man als Frameworkverwender entscheiden, welche konkreten Implementierungen eingefügt (injected) werden sollen, ob diese beispielsweise Singletons sind. Zudem bietet es die Möglichkeit, Proxies um die Objekte zu bauen und somit Ansätze von Aspektorientierung einzubauen. Um nun ein Objekt injected zu bekommen, lassen sich bei objektorientierten Systemen drei mögliche Injectionarten unterscheiden.

- **Konstruktorinjection**: Alle benötigten Objekte werden im Konstruktor übergeben. Dies hat den Vorteil, dass bei Tests kein Objekt vergessen werden kann.
- **Methodinjection**: Es werden Methoden aufgerufen, bei denen die Objekte übergeben werden. Der Vorteil dabei ist, dass man dem Objekt nicht immer alle Abhängigkeiten geben muss, sondern nur diejenigen, die es bei dem Test benötigt.
- **Attributinjection**: Die Attribute werden direkt gesetzt.

Es gibt es vielerlei Frameworks, die diese Aufgabe übernehmen können. Die bekanntesten sind Spring[4] und Guice[5]. Guice ist ein sehr leichtgewichtiges Framework [Vano8], sodass es ohne viel Aufwand in bestehende Systeme integrierbar ist. Dies und die sehr einfache API, waren der Grund für die Verwendung von Guice für diese Arbeit. Dabei wird die @Inject-Annotation über den Konstruktor, den Methoden, bzw. des Attributs verwendet, um anzudeuten, dass Guice die Objekte beim Erstellen von Objekten dieser Klasse injecten soll. Will man mehrere Objekte vom gleichen Typ, so lässt man sich einen Provider injecten, über den man diese Objekte dann holen kann. Dies bietet sich auch an, wenn erst zur Laufzeit Objekte benötigt werden.

[3] http://de.wikipedia.org/w/index.php?title=Dependency_Injection&stableid=85627713
[4] http://www.springsource.org/
[5] http://code.google.com/p/google-guice/

3 Konzepte

3.1 Datenmodell

Daten lassen sich in konkrete und kombinierte Daten einteilen. Die konkreten Daten, wie „Windrichtung“ und „Windgeschwindigkeit“, enthalten die einfache Werte. Die kombinierten Daten, wie beispielsweise „Winddaten“, bestehen aus Kombinationen dieser konkreten Daten.

Ein weiterer Unterschied zwischen Daten liegt in ihrem optimalen Wert. Während die Einen ihr Optimum bei einem Minimum oder Maximum haben, haben dies die Anderen mitten in der Skala. Dies hat den Effekt, dass sowohl zu niedrige, als auch zu hohe Werte für den Anwender schlecht sind. Dies ist zum Beispiel bei der Temperatur der Fall.

3.1.1 Ampelfähigkeit

Daten können abhängig von der Aktivität und des Benutzers in bestimmte Kategorien eingeteilt werden. Eine solche Einteilung kann über eine Normalisierung gemacht werden, die die ursprünglichen Werte in einen Indexwert umwandelt. Für diese Zuordnung können Richtlinien, wie beispielsweise die EU-Richtlinie für Luftreinheit [1], verwendet werden.

Zudem kann das System diese Grenzwerte an die auszuführende Aktivität und deren Anwender anpassen. Darunter fällt unter anderem der Gesundheitszustand des Nutzers. Ist er auf bestimmte Pollen allergisch, so können die Grenzwerte gezielt heruntergesetzt werden. Auch bei Wanderrouten kann je nach Alter und Fitness die Wanderroute anders bewertet werden.

In dieser Arbeit wurde ein Index verwendet, der drei Zustände hat: Gut, Mittel und Schlecht bzw. Unbedenklich, leicht Bedrohlich und Gesundheitsgefährdend. Da dies an eine Ampel erinnert, wird dies hier als „Ampelfähig“ bezeichnet.

[1] http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2008:152:0001:0044:DE:PDF

3.1.2 Normalisierung

Die Normalisierung von ampelfähigen Daten wird in dieser Arbeit mittels eines Index-Wertes gelöst, der wie folgt aufgebaut wird:

$0 = \text{Index}$	Optimaler Wert
$0 < \text{Index} <= 1$	Gut
$1 < \text{Index} <= 2$	Mittel
$2 < \text{Index}$	Schlecht

Dabei wird zwischen $0 \ldots 1$ und $1 \ldots 2$ interpoliert. Bei den Werten > 2 wird die Steigung des 1..2 Bereiches verwendet. Zudem wird für einige Darstellungen der Wert 3 als Maximum verwendet.

3.1.3 Datenarten

Für die Anwendungen in dieser Arbeit sind hauptsächlich Aktivitäten unter freiem Himmel für den Benutzer von Interesse. Deshalb wurde hierauf der Fokus gelegt. Ein Aspekt ist hierbei die **Temperatur**. Diese hat die Besonderheit, dass sowohl zu niedrige, als auch zu hohe Temperaturen gefährlich sein können. Dies muss bei der zuvor in Kapitel 3.1.2 ausgeführten Normalisierung beachtet werden. Ein weiterer wichtiger Aspekt bei Aussenaktivitäten ist der **Wind**. Dieser gliedert sich in Windrichtung und Windstärke. Das besondere hierbei ist, dass eine recht hohe Windstärke bei der Aktivität nicht von Nachteil sein muss. Will man beispielsweise Fahrradfahren, so freut man sich eventuell sogar über einen möglichst starken Rückenwind. Eine ähnliche Eigenschaft hat die **Bodenhöhe**, welche bei einer Route die Steigung bzw. das Gefälle ist. Auch die **Pollenbelastung** ist für einige Menschen sehr wichtig, wenn es um das Planen von Außenaktivitäten geht. Ist man stark allergisch, so muss dies natürlich bei der Darstellung berücksichtigt und kritische Werte hervorgehoben werden. Ein weiterer Aspekt, der zu beachten ist, ist die **Luftqualität**, worunter unter anderem Feinstaub und Ozonwerte fallen. Mit möglichen Folgeschäden und einigen Grenzwerte hat sich [LS05] beschäftigt. Diese Luftqualitätswerte haben meist die Eigenschaft, dass Werte nahe am Nullpunkt optimal sind. Je höher die Werte werden, desto schlechter wird somit die Luftqualität. Ein ebenfalls sehr wichtiger Aspekt ist die **Wetterlage**, wie beispielsweise Regen, Schneefall oder Sonne.

3.2 Unsicherheit

Fast alle Daten sind mit Unsicherheit behaftet. Um fundierte Entscheidungen treffen zu können, müssen die Auswirkungen der Unsicherheit den Analysten bekannt sein. Falls die Unsicherheit in den Daten ignoriert oder missverstanden wird, kann dies zu falschen Entscheidungen führen [Riv07]. Woher diese Unsicherheit kommt, und in welche Gruppen diese in dieser Arbeit eingeteilt werden, behandelt dieses Kapitel.

3.2.1 Mögliche Ursachen von Unsicherheit

Unsicherheit kann viele ganz unterschiedliche Faktoren beinhalten. Dazu zählen die Störungen der Hardware, Unsicherheiten durch mehrere Anbieter, gemittelte Werte oder Ungenauigkeiten, die durch die Darstellungen entstehen.

Da die Daten mittels **Sensoren** gemessen werden, ist bereits deren Auflösung und eingeplanten Messungsungenauigkeit relevant. Zudem gehören Rauschen und andere Störungen, die die Datenqualität reduzieren. Auch können einzelne Sensoren beschädigt sein und so ungenaue oder falsche Daten liefern.

Falls es für ein Datum **mehrere Datenanbieter** gibt, können für dieses Datum unterschiedliche Messwerte vorliegen. Auch kann an Datenvorhersagen von Anbietern eher gezweifelt werden, wenn diese in der Vergangenheit falsche Daten lieferten. Allgemein lässt sich sagen, dass heutzutage bei Wetterdaten, die die Zukunft betreffen, immer ein gewisser Grad an Unsicherheit besteht. Auch wenn Daten ungewöhnliche Ausschläge aufweisen, ist an diesen Daten möglicherweise mehr zu zweifeln als an realistischeren Werten.

Ein weiteres Problem besteht in der Datenmenge. Die Daten der Sensoren werden häufig über einen Zeitraum **gemittelt** oder in einem Bereich zusammengefasst. Beispielsweise werden Wetterdaten üblicherweise in Stunden oder sogar Tagen gemittelt. Zu dieser zeitlichen Komponente kommt noch hinzu, dass die Sensoren nur an bestimmten Orten aufgestellt sind. Die Daten dazwischen werden dann interpoliert, was zu weiteren Ungenauigkeiten führt.

Am Ende werden die Daten dargestellt, wobei es zu weiteren Ungenauigkeiten kommt. **Darstellungen** reduzieren die Genauigkeit meist ebenfalls. Durch Normalisierung und anderen Transformationen wird dieses Problem noch verschärft. Darstellungen vermitteln immer auch einen subjektiven Eindruck der Daten, was die Wahrnehmung beeinflussen kann. Auch besteht die Gefahr, dass es sich teilweise um sehr komplexe Datensätze handelt, die falsch interpretiert werden.

Zusammenfassung und Quellen:

- Messfehler (Hardware)

  Messungsungenauigkeit [Riv07]

  Auflösung der Sensoren [Riv07] [GS06] [THM+05]

  Störungen (Rauschen) [GS06] [KKEM10]

  Beschädigte Sensoren

- Unsicherheit

  Mehrere unterschiedliche Messwerte [Riv07]

  Verlässlichkeit (schlechte Erfahrungen mit Datenanbieter) [Riv07]

  Zukunftsvorhersagen (Statistiken, Wetter-Berechnungen)

Zweifelhafte Daten (Ausschläge zu vergangenen Daten) [Riv07] [THM+05]

- Präzision

  Durchschnittsbildung für Zeit/Ortsintervall

  Bereichsbildung (von-bis) [KKEM10]

  Interpolation zwischen Messpunkten [KKEM10] [THM+05]

- Darstellung

  Schlechte Darstellung [THM+05]

  Transformationen [THM+05]

  Subjektive Eindrücke [GS06]

  Zu komplexe Daten [THM+05]

3.2.2 Verwendung in dieser Arbeit

Für diese Arbeit wird, wie in [OM02], die Unsicherheit in eine statistische Unsicherheit und eine Bereichsunsicherheit aufgetrennt. Dabei wird die Bereichsunsicherheit den Bereich angeben, indem der Wert liegen wird. Über die statistische Unsicherheit kann zusätzlich noch angegeben werden, mit welcher Wahrscheinlichkeit der Wert außerhalb dieses Bereiches liegen kann. Dies kann durch die Ursachen, die im vorherigen Kapitel 3.2.1 beschrieben wurden, nötig sein.

Im Datenmodell wird die Unsicherheit folgendermaßen abgebildet: Die statistische Unsicherheit wird mittels eines separaten Attributs realisiert, das einen Wert zwischen 0 (0% sicher) und 1 (100% sicher) annehmen kann. Der eigentliche Wert wird mittels eines „min“ und eines „max“ Wertes realisiert. Falls eine Komponente nur einen konkreten Wert darstellen kann, so wird bei dieser der Mittelwert verwendet.

3.3 Darstellungen

Mit dem im Abschnitt 3.1.3 definierten Datenmodell kann nun die Darstellung der Daten beschrieben werden. Dazu werden verschiedene Darstellungen untersucht und in Kategorien eingeteilt. Dabei wird vor allem auf die darstellbaren Daten und die Möglichkeit zur Einbindung von Unsicherheit eingegangen.

3.3.1 Darstellungsarten

Darstellungen lassen sich in Fläche, Route Punkt und Separate Darstellung einteilen, die jeweils spezielle Vor und Nachteile aufweisen:

Fläche

Es gibt Darstellungen, die Daten auf einer Fläche anzeigen. Diese werden üblicherweise in einem Rechteck um den interessanten Teil („Regions of Interest“ [BDW+08]) dargestellt. Der Nachteil an dieser Methode ist, dass die Darstellung meist nur Daten von einem konkreten Zeitpunkt gleichzeitig anzeigen kann. Dafür erhält man aber einen guten Eindruck dieses Zeitpunktes. Man kann in dem aktuell angezeigten Zeitschritt beispielsweise Minima, Maxima oder Häufungen erkennen. Will man hier auch die zeitliche Komponente berücksichtigen, benätigt man meistens eine Animation über die Zeit. Dabei ändert man in regelmäßigen Abständen den Zeitpunkt der dargestellten Daten.

Route

Bei den meisten Anwendungsfällen soll eine Route analysiert werden. Darstellungen, welche die Daten entlang der kompletten Route anzeigen, bilden die zweite Kategorie. Hier lassen sich die Werte, die zu den Zeitpunkten, zu denen man sich an dem Punkt der Route befindet, ablesen. Dies bietet einen guten Überblick über die Daten der komplette Route, ohne dass eine Animation notwendig ist.

Punkt

Die nächste Möglichkeit ist die Darstellung an genau einem Punkt der Route. Dabei wird an der Stelle gezeichnet, wo sich der Datenpunkt befindet. Um alle für die Route relevanten Informationen zu erhalten, wird wie bei der Flächendarstellung auf Animationen zurückgegriffen werden.

Separate Darstellung

Es besteht auch die Möglichkeit einen Datenpunkt unabhängig von der Karte darzustellen. So können zusätzliche Informationen angezeigt werden, ohne dass diese andere Darstellungen überdecken. Ansonsten sind hier die selben Einschränkungen zu beachten wie bei der Punkt-Darstellung direkt auf der Karte.

Aktuelle Position

Um den Anwender nicht mit mehreren unterschiedlichen Zeitpunkten zu sehr zu verwirren, wird es nur genau einen aktuellen Zeitpunkt geben. Dieser lässt sich vom Benutzer einstellen oder animieren. Die Flächen-, Punkt- und Separate-Darstellung werden sich an diesem aktuellen Zeitpunkt orientieren.

3.3.2 Darstellungen allgemein

In folgenden Kapitel wird auf die die grundsätzlichen Eigenschaften eingegangen, die für alle Darstellungen gelten.

Farben

Farben bieten grundsätzlich eine gute Möglichkeit, die Aufmerksamkeit des Anwenders auf ein wichtiges Detail zu lenken. Signalfarben wie beispielsweise Rot, werden häufig für Gefahr verwendet 2.1.1. Ampelfähige Daten 3.1.1 lassen sich also besonders gut mit Farben darstellen.

Auch muss darauf geachtet werden, dass beispielsweise die sehr weit verbreitete Rot-Grün Blindheit, oder andere Farbfehlsichtigkeiten 2.1.2, nicht zum Problem werden.

Will man vielerlei unterschiedliche Darstellungsmöglichkeiten mit Farben gleichzeitig einsetzen, so kann dies schnell unübersichtlich werden. Aus diesem Grund sollten Farben gezielt und sparsam verwendet werden.

Animationen

Animationen sind eine gute Möglichkeit, um eine zusätzliche Dimension der Daten darzustellen. Beispielsweise kann man, wie bei der Flächendarstellung in Kapitel 3.3.1 beschrieben, die zeitliche Komponente damit abbilden. Animationen können aber auch für die Darstellung von Bereichs-Unsicherheit eingesetzt werden. Statt nur den Mittelwert anzuzeigen, kann dieser Wert zwischen Minimum und Maximum schwanken. Das Problem hierbei ist, dass es für den unerfahrenen Benutzer so aussehen kann, als ob vorhergesagt wird, dass der Wert sich schnell an dieser Stelle ändert [HT06].

Zudem kann schlecht über die Zeit und die Unsicherheit gleichzeitig animiert werden, da für das Ablesen der Unsicherheit mit Animationen etwas Zeit benötigt wird, in der sich die Werte längst durch die zweite Zeit-Animation verändert haben. Beispielsweise, wenn Unsicherheit über die Windrichtung mit einem sich drehenden Pfeil dargestellt wird, muss die Animation der Pfeildrehung einmal durchlaufen werden, da man sonst die Werte nicht erkennen kann. Auch kann dies schnell zu Fehlinterpretationen führen, wenn die Drehung des Pfeils mit der Animation über die Zeit in Verbindung gebracht wird.

Ein weiterer Nachteil ist, dass sich Animationen nicht, für zum Beispiel eine Wanderkarte, ausdrucken lassen [HT06].

Transparenz und Schärfe

Transparenz wird oft für Unsicherheit eingesetzt [Riv07] [GS06]. Dabei wird das Objekt bei steigender Unsicherheit transparenter dargestellt. Der Vorteil von Transparenz liegt darin, dass die Daten umso schlechter ablesbar sind, je unsicherer der damit verbundene Wert ist.

Neben Transparenz erfüllt auch Unschärfe [CR00] diese Eigenschaft und ist somit ebenfalls eine Möglichkeit, die Unsicherheit darzustellen.

Linien

Sind in einer Darstellung Linien vorhanden, so kann dort mittels mehrerer Methoden die Unsicherheit dargestellt werden: Zum Einen können **Farben** verwendet werden. Dies ist aufgrund der eventuell schmalen Linien und des eventuell ebenfalls eingefärbten Untergrundes nicht zu empfehlen. Je unsicherer die Daten sind, desto mehr kann man die Linie auch **verpixeln**. Dazu malt man die Punkte der Linie je nach Unsicherheit versetzt zur eigentlichen Position [Mac92]. **Störungen** einzubauen [CR00] hat dabei einen ähnlichen Effekt. Eine weitere Möglichkeit ist es, die Linie nicht gerade, sondern **gezackt** zu zeichnen. Dabei wird je nach Unsicherheit die Linie größere Ausschläge aufweisen [Mac92]. Auch lassen sich andere Attribute einer Linie anpassen, wie beispielsweise die **Liniendicke** [HBO10] [WSF+95] oder **Transparenz** 3.3.2.

Welche Methode für die Darstellung sinnvoll ist, hängt vom Anwendungsfall ab. Ist die Linie beispielsweise Teil eines Glyphen, so ist es eventuell eher sinnvoll, den ganzen Glyphen unscharf oder verpixelt darzustellen[WSF+95]. Ist die Linie jedoch eigenständig, wie in einem Liniendiagramm, kann man auch statt der Linie eine ausgefüllte Fläche zwischen den minimalen und maximalen Werten darstellen. Wird hingegen eine Grenze zwischen Staaten dargestellt, so kann die gezackte oder gestörte Variante sinnvoll sein.

Größe

Bei vielen Darstellungen kann ein Größenparameter eingestellt werden. Sei es die maximale Größe der Icons, oder der maximale Abstand zur Route. Diese Größe soll der Anwender modifizieren können um die Daten besser ablesbar zu machen oder um einen besseren Überblick über das Gesamtbild zu erhalten.

Es lassen sich grundsätzlich zwei verschiedene Zoom-Möglichkeiten unterscheiden: Eine ist das **Zoomen in die Karte**, um mehr Daten auf einem bestimmten Teil der Route zu sehen. Dies wird von dem Kartenframework selbst unterstützt. Dabei wird aber nur die Auflösung, nicht aber die Größe der Darstellungen erhöht. Die Icons sind einem Zoomen in die Karte nicht größer, sondern der Geo-Abstand der Icons wurde verringert. Das Ändern der **Darstellungsgröße** ist somit die zweite Möglichkeit zu zoomen. Diese ändert ebenfalls

die Auflösung: Je größer die Icons, desto weiter müssen diese voneinander entfernt sein und desto geringer ist somit die örtliche Auflösung.

Das Zoomen der Karte ist unerlässlich, um genauere Informationen über Teile der Route zu erhalten. Das Zoomen der Icons sollte ebenfalls in einem solchen Framework enthalten sein, um die Größe der Glyphen an den Benutzer anpassen zu können.

Beschriftungen

Es stellt sich die Frage, wie man Darstellungen beschriften kann, sei es die Art (Bsp.: „Windstärke“) oder den konkreten Wert („30km/h“). Hierzu gibt es allerlei Möglichkeiten:

Die Einfachste Möglichkeit ist meist eine Legende einzufügen und diese mittels Farben mit dem Datensatz zu kombinieren. Diese hat den Nachteil der bereits in Kapitel 3.3.2 beschrieben wurde.

Mittels Einfügen von zusätzlichen Texten an die Datenstellen, kann diese Information ebenfalls hinzugefügt werden. Man kann beispielsweise, falls die Darstellung Flächen zeigt, den Namen des hier dargestellten Wertes einfügen. Bei Linien können diese Beschriftungen ebenfalls gut angebracht werden. Nachteil davon ist, dass diese andere Informationen überdecken können.

Eine elegantere Möglichkeit ist das dynamische Anzeigen solcher Beschriftungen [EKHW07]. Dies lässt sich gut mittels Mousehover realisieren, bei dem neben der Mausposition Tooltips [2] verwendet werden, in denen entsprechende zusätzliche Informationen abgebildet sind [BW08].

Zur Unterstützung der Darstellung kann das Element, zu dem diese Information gehört, noch hervorgehoben werden. Diese temporäre Anzeige hat den Vorteil, dass hierbei mehr Informationensdetails in der Darstellung untergebracht werden können. Nachteil dieser Methode ist, dass diese wie die Animation 3.3.2 nicht gedruckt werden kann.

3.3.3 Heatmap

Eine Heatmap ist eine flächendeckende Einfärbung der Fläche (Abbildung 3.1). Diese gehört somit in die Kategorie der Flächen-Darstellungen 3.3.1 und verwendet nur Farben für die Darstellung der Werte.

Besonders gut ist eine Heatmap für Temperaturdaten geeignet. Auch ampelfähige Daten 3.1.1 lassen sich, durch den starken Einsatz von Farben 3.3.2, sehr gut darstellen. Zudem können Heatmaps sehr gut sehr dichte und wohldefinierte Daten darstellen [Fis07].

[2] http://de.wikipedia.org/w/index.php?title=Tooltip&stableid=88401779

Um eine Heatmap zu erstellen wird ein Raster an Datenpunkten benötigt. An den Punkten werden nun die Farben für diese Positionen berechnet. Zwischen den Punkten wird dann die Farbe interpoliert bzw. extrapoliert [Fis07].

Unsicherheit lässt sich über verschiedene Möglichkeiten in eine Heatmap einbauen. Zum Einen können die Daten mit einer zusätzlichen Dimension versehen werden. Bei dieser werden die Farben dann zum Beispiel transparenter 3.3.2 dargestellt. Eine weitere Idee ist das Bleichen der Farbe [Riv07]. Hierbei wird die Farbe ins Weiße verschoben. Weiß wird dabei als fehlende Information interpretiert [HT06]. Ein sichtbares Raster über die Heatmap zu legen, ist eine weitere Möglichkeit [CR00]. Dies wird mittels Linien um die Datenpunkte realisiert. Mit diesem Raster kann dann mittels den in Kapitel 3.3.2 genannten Methoden Unsicherheit eingefügt werden.

Das Animieren über die Zeit ist für Heatmaps besonders gut geeignet. Man kann damit sehr schön den Verlauf der Werte während der Animation erkennen und sieht Tendenzen sehr schnell. Leider lassen sich auch nur diese Tendenzen und grobe Werte ablesen. Zudem kommt die Problematik der Farbfehlsichtigkeiten vieler Menschen 2.1.2, die durch die alleinige Darstellung mittels Farben hier besonders kritisch ist.

Eine Heatmap erfordert eine Legende, in der beispielhaft die Farben und deren Bedeutung dargestellt werden. Um diese Daten genauer ablesen zu können, kann man die Beispieldaten, wie in Abbildung 3.2 gezeigt, mittels eines Farbverlaufes darstellen. Falls Unsicherheit mit angezeigt wird, kann hier eine zweidimensionale Legende verwendet werden [HT06].

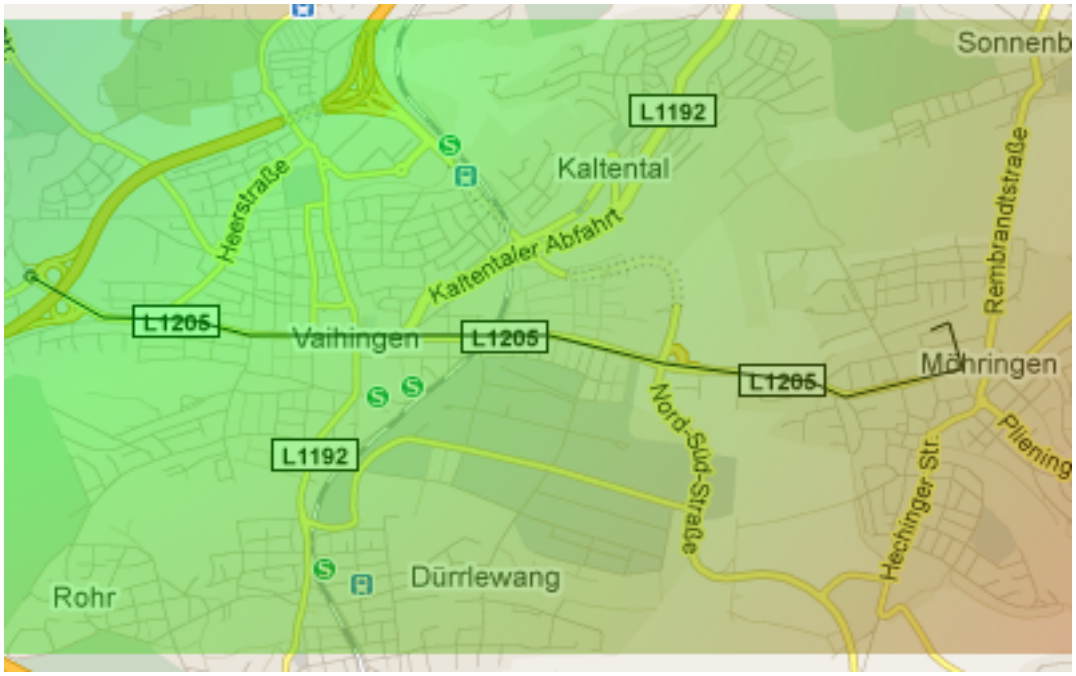

Abbildung 3.1: Darstellung einer **Heatmap** in der Beispielanwendung.

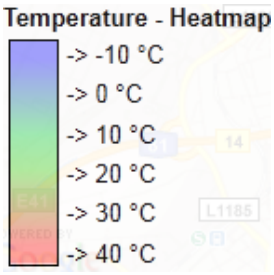

Abbildung 3.2: Legende der Heatmap mit Farbverlauf.

3.3.4 Isolinien

Isolinien sind Linien, die Wertebereiche abgrenzen (Abbildung 3.3). Aus Isolinien entstehen Flächen, in denen die Messwerte alle in einem bestimmten Bereich liegen. Isolinien zählen wie Heatmaps ebenfalls zur Flächendarstellung 3.3.1. Ein schon langes übliches Einsatzgebiet von Isolinien sind Höhenlinien auf statischen Karten.

Es lassen sich bei dieser Darstellung die gleichen Datensätze verwenden wie mit Heatmaps. Zudem können Höhenwerte sehr intuitiv dargestellt werden.

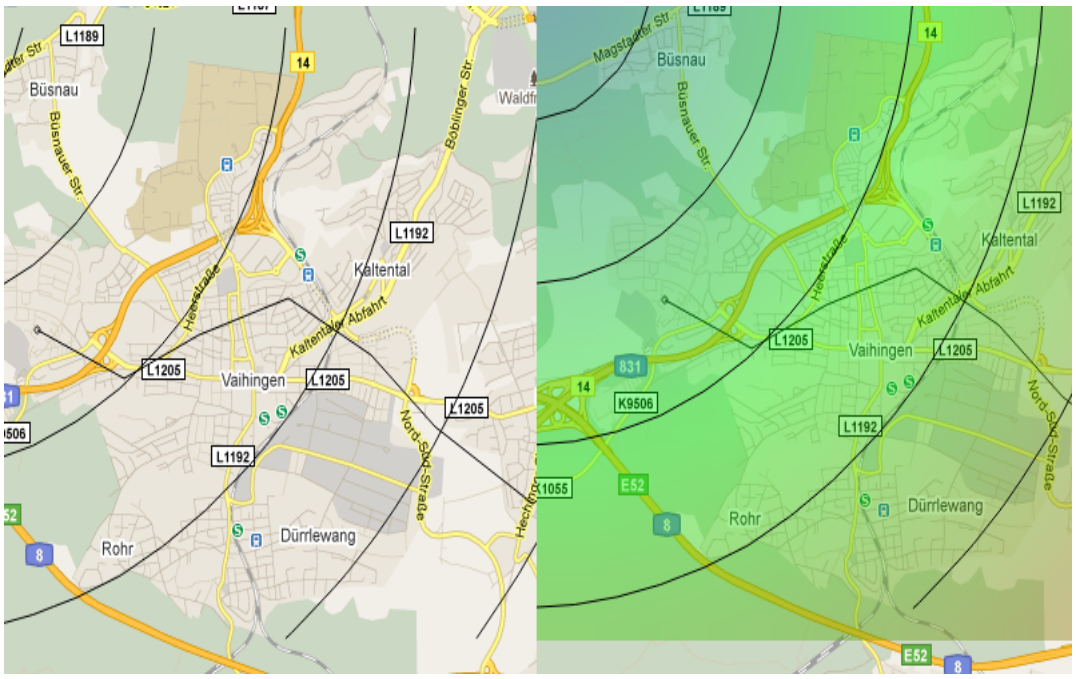

Abbildung 3.3: Im linken Bild sind **Isolinien** in der Beispielanwendung dargestellt. Diese können unter anderem, wie im rechten Bild gezeigt, zur Unterstützung einer Heatmap verwendet werden. Die Isolinien dienen hierbei dazu, die Wertebereiche innerhalb der Heatmap deutlich abzugrenzen.

Bei Isolinien wird versucht, entlang eines konkreten Wertes Strecken einzufügen. Hierbei geht man wie auch bei Heatmaps von einem Raster an Datenpunkten aus und fügt an den jeweiligen Stellen die Linien ein, so dass diese die geforderten Strecken bilden. Ein Algorithmus zur Erstellung solcher Linien wird in [LP09] beschrieben.

Unsicherheit lässt sich in Isolinien nur schwer darstellen. Die statistische Unsicherheit kann man, mit den in Kapitel 3.3.2 beschriebenen Methoden, darstellen. Die Bereichsunsicherheit dagegen ist aufgrund von möglichen Überlappungen der Flächen nicht intuitiv mit dieser Ansicht darzustellen.

Isolinien sind sehr gut dazu geeignet, um Heatmaps zu ergänzen. Mit dieser Erweiterung lassen sich die Daten der Heatmap besser zuordnen und Verläufe viel besser erkennen (Abbildung 3.3).

3.3.5 Windpartikel

Für die Darstellung von Wind auf einer Karte können Windpartikel verwendet werden. Dies sind Teilchen, die sich je nach Wind über die Karte bewegen (Abbildung 3.4). Dabei wird zu

Abbildung 3.4: Darstellung von **Windpartikeln** in der Beispielanwendung. Man sieht bei dieser Darstellung schnell, woher der Wind kommt und wie stark er ungefähr ist. Selbst auf statischen Bildern, wie hier gezeigt, kann man die Windrichtungen an dem Vektor und Farbverlauf der Linien erkennen. Die Windstärken lassen sich an der Länge der Linien ablesen.

den Teilchen jeweils eine Pathline 2.2 angehängt, um die früheren Positionen darzustellen. Es handelt sich hierbei ebenfalls um eine Flächendarstellung 3.3.1.

Diese Darstellungsart ist speziell für Daten geeignet, die eine Richtung und eine Geschwindigkeit aufweisen.

Es werden Strecken auf die Karte gemalt und bei jeder Änderung wird das hinterste Element der Strecke entfernt und am Anfang angefügt. Die Richtung und Länge der angefügten Strecke hängt von dem Datenpunkt ab. Nebeneffekt dieser Historie ist, dass man die Geschwindigkeit des Partikels gut erkennen kann: Je länger die Pathline ist, desto höher ist die Windgeschwindigkeit. Zudem ist es sinnvoll, dass man die aktuelle Position kennzeichnet. Dies fördert die Lesbarkeit und sorgt zudem noch dafür, dass sich auch bei ausgedruckten Karten diese Information gut ablesen lassen. Dies kann zum Beispiel mit Transparenz erreicht werden, wobei der aktuelle Punkt ganz sichtbar und der letzte Punkt vollständig transparent ist. Nutzt man die Transparenz anderweitig, kann man auch einen Punkt oder ein Pfeil als Partikelspitze verwenden.

Da es sich hierbei um Strecken handelt, lässt sich hier alles anwenden, was in Kapitel 3.3.2 beschrieben ist. Zudem können die Positionen der nächsten Streckenabschnitte zufällig in dem Bereich eingesetzt oder dies animiert werden. Dies ist aber problematisch, da bei hoher Bereichsunsicherheit die Partikel sich in einem Zickzack-Kurs über die Karte bewegen oder Partikel ihre Wege kreuzen könnten. Da sich die Partikel in der Realität anders verhalten ist dies nicht zu empfehlen.

Diese Darstellung ist sehr intuitiv, da diese die Auswirkungen von Wind am Beispiel von Partikeln zeigt. Zudem lässt sich diese Darstellung sehr gut parallel zu vielen anderen Darstellungen einsetzen, da sie weder Farben verwendet noch großflächig andere Elemente überdeckt.

3.3.6 Stackedgraph

Ein Stackedgraph ist ein Graph, bei dem mehrere Datenschichten übereinandergelegt sind (Abbildung 3.5). Diese Art von Graph hat den Vorteil, dass sowohl die Summe aller Einzeldaten als auch deren Verläufe gut ablesbar sind [BW08]. Dieser wird normalerweise auf einer festen geraden Ausgangslinie gezeichnet. In unserem Fall bietet es sich an, für die Grundlinie die Route zu verwenden. Somit erhält man eine Routen-Darstellung 3.3.1 der Daten.

Diese Darstellung ist nur für Daten geeignet, die addiert werden können. Sie müssen also die gleiche Einheit und die gleiche Bedeutung haben. Dies ist besonders gut für die Darstellung von Gesamtmengen, die in Teilmengen oder Kategorien aufgeteilt werden können, geeignet.

Bei einem Stackedgraph wird zunächst auf die x-Achse der erste Datensatz gezeichnet. Diese neue Linie dieses Datensatzes wird dann als Grundlinie für den nächsten Datensatz verwendet. Wenn man als erste Grundlinie nun keine gerade Linie, sondern eine Route,

bestehend aus kurzen Einzellinien, hat, muss man noch ein paar Besonderheiten bei den Eckpunkten beachten, die im Implementierungskapitel 4.4.7 behandelt werden.

Bereichsungenauigkeit lässt sich schwer mit dieser Darstellung anzeigen, da die Positionen der oben liegenden Schichten alle von den darunter liegenden abhängen. Zeigt man die Bereichsunsicherheit mittels eingefügten Glyphen [OM02], so lässt sich das Minimum und Maximum der Gesamtsumme nicht ablesen. Da die Unsicherheiten der Einzeldaten unterschiedlich sein können, können zu viele Kombinationen auftreten, die auch mittels Animation bei einem Stackedgraph nicht alle dargestellbar sind. Es kann zwar wie in Kapitel 3.3.2 beschrieben zwischen Minimum und Maximum aller Daten animiert werden, dennoch sind hier dann nur Best- und Worstcase-Szenarien abbildbar.

Es gibt kaum Umweltdaten, die addierbar sind. Luftqualität beispielsweise besteht, wie in Kapitel 3.1.3 erwähnt, aus vielen Daten, die aber alle unterschiedliche Grenzwerte und Einheiten haben. Auch Pollenbelastung lässt sich nicht sinnvoll darstellen. Zwar haben diese alle die gleiche Einheit, dafür sind die Auswirkungen der einzelnen Pollenarten für den Anwender ganz unterschiedlich. Ist er auf einzelne Pollen allergisch, ist ihm nur diese Konzentration wichtig.

Abbildung 3.5: Darstellung eines **StackedGraph** in der Beispielanwendung mit zwei Werten. Zusätzlich ist der Stackedgraph noch an der Route gespiegelt.

3.3.7 Liniendiagramm

Die Messpunkte werden bei Liniendiagrammen durch Linien verbunden (Abbildung 3.6). Wie schon beim Stackedgraph 3.3.6 wird hier als Basislinie die Route verwendet, um die zeitliche und örtliche Komponente gleichzeitig darzustellen. Ein Liniendiagramm unterstützt zudem die Darstellung mehrerer unterschiedlicher Messwerte.

Im Gegensatz zum Stackedgraph 3.3.6 müssen die Werte in einem Liniendiagramm nicht addierbar sein. Es können mehrere Skalen verwendet werden um diese gleichzeitig anzuzeigen. Für unseren Fall ist es aber sinnvoller, diese Daten zu normalisieren, um eine einheitliche Skala zu schaffen. Hier bieten sich vor allem ampelfähige Daten 3.1.1 an. Ist ein Wert gefährlich, so sticht er sofort hervor. Damit lassen sich einfach die gefährlichen bzw. die schlechten Werte schnell erkennen.

Das Diagramm wird wie ein Stackedgraph 3.3.6 aufgebaut. Es können hierbei die gleichen Methoden verwendet werden. Zusätzlich ist es sinnvoll, Hilfslinien anzuzeigen, um beispielsweise bei den ampelfähigen Daten 3.1.1 Gut/Mittel und Schlechte Werte erkennen zu können. Dabei ist es sinnvoll, die durch die Hilfslinien abgetrennten Bereiche einzufärben, um sie besser zu erkennen.

Da es sich hierbei um Linien handelt, lassen sich die in Kapitel 3.3.2 beschriebenen Methoden verwenden, um Unsicherheit darzustellen. Die Linie kann auch in eine Fläche umgewandelt werden, um Bereichsunsicherheit darzustellen [THM+05].

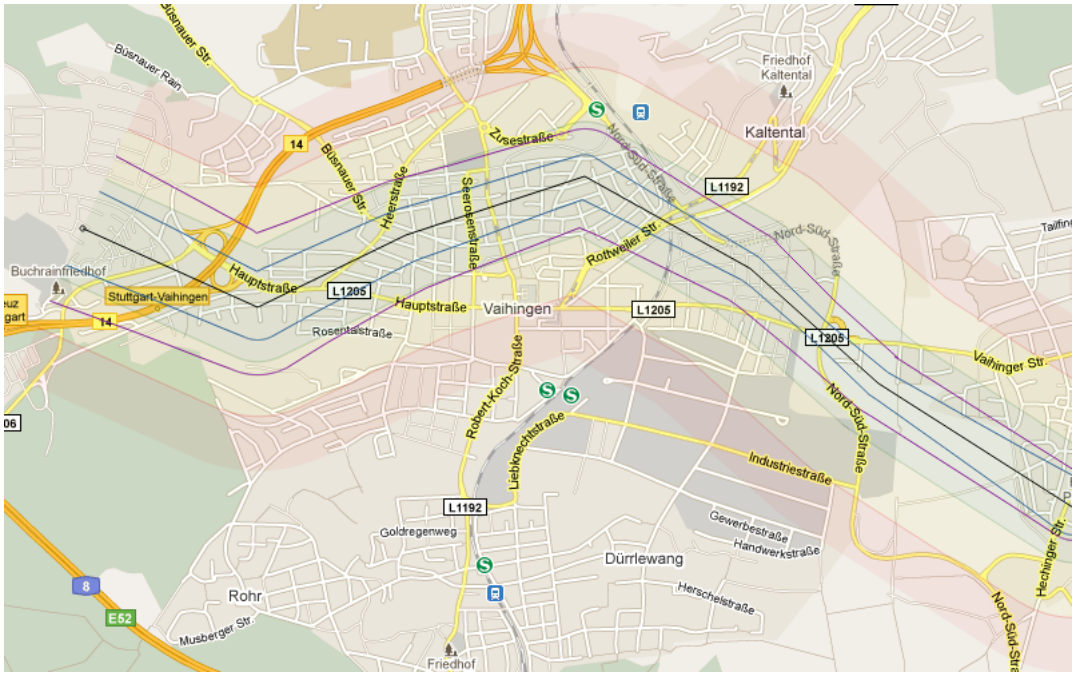

Abbildung 3.6: Darstellung eines **Liniendiagramms** in der Beispielanwendung mit zwei Wertelinien (Lila und Blau).

3.3.8 Glyphen allgemein

Eine weit verbreitete Darstellungsmöglichkeit sind Glyphen, die auf der Karte platziert werden. Glyphen lassen sich in allen Darstellungsarten gut einsetzen. In Flächendarstellungen lassen sie sich als Raster über die Karte legen. In Routen-Darstellungen können sie nebeneinander auf der Route platziert werden. Ideal geeignet sind sie auch in Punkt oder Separate Darstellungen.

Da fast alle Glyphen aus Linien aufgebaut sind, können hier die in Kapitel 3.3.2 beschriebenen Methoden gut angewendet werden. Bei statischen Glyphen lassen sich auch Animationen 3.3.2 einfügen, um die Unsicherheit zu visualisieren.

Hinzu kommt die Möglichkeit, die Icons unterschiedlich groß darzustellen. Je größer das Icon, desto größer ist der von ihm repräsentierte Wert. Dies hat den Vorteil, dass kleine Werte nicht so sehr hervorstechen wie größere Werte. Dies ist vor allem sinnvoll, wenn große Werte gefährlich oder anderweitig wichtig sind.

3.3.9 Uhr

Um die Uhrzeit an einem bestimmten Punkt zu erkennen, kann eine Uhr dargestellt werden (Abbildung 3.7). Dieser Glyph lässt sich nicht für die Flächendarstellung einsetzen, da dort überall die gleiche Uhrzeit an einem Zeitpunkt ist.

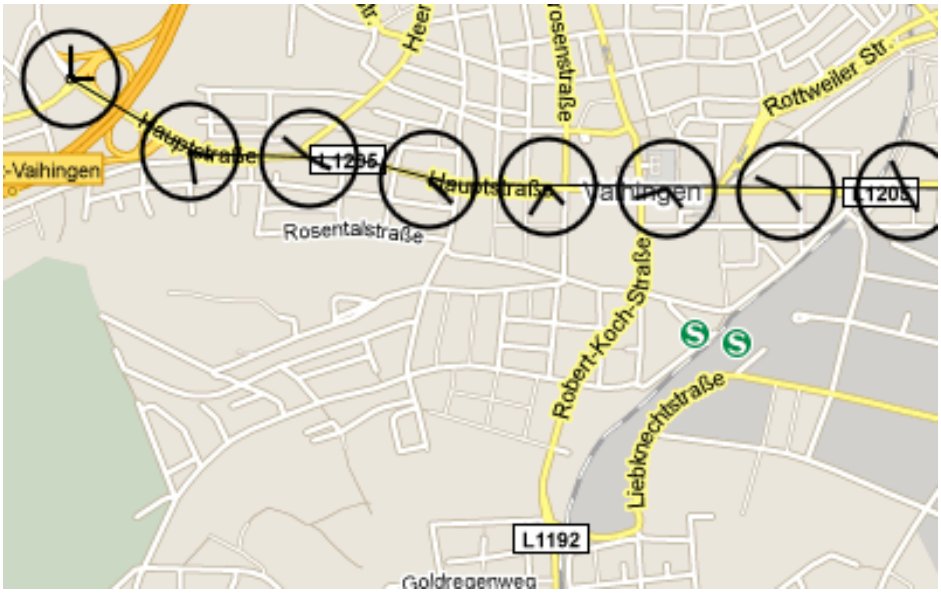

Abbildung 3.7: Darstellung einer Uhr in der Beispielanwendung entlang einer Route. Dabei befindet man sich jeweils zu den dargestellten Uhrzeiten an den jeweiligen Orten.

3.3.10 Texte

Eine Darstellung, die fast überall einsetzbar ist, ist das Texticon (Abbildung 3.8). Diese hat den Nachteil, dass dieser Glyph eine variable Breite haben kann. Falls zu viel Text vorhanden ist, kann dies schnell für den Benutzer unübersichtlich werden.

Abbildung 3.8: Darstellung von **Texticons** in der Beispielanwendung, anhand der Temperaturen, die zu den Zeitpunkten erwartet werden.

3.3.11 Windpfeil

Ein Windpfeil ist die gebräuchliche Darstellung von Wind, wie man sie oft auf Wetterkarten findet (Abbildung 3.9). Dieser kann sehr intuitiv die Windrichtung darstellen. Die Windstärke dagegen lässt sich hiermit nur relativ zu anderen Windpfeilen darstellen. Eine Möglichkeit ist die Größe des Glyphen zu verwenden. Je größer der Glyph gezeichnet wird, desto stärker ist der Wind an der Stelle. Dies hat den Vorteil, dass starke Winde besser gesehen werden als schwache. Eine weitere Möglichkeit, die Windstärke darzustellen, ist eine Linie oder ähnliches innerhalb des Pfeils zu animieren (Abbildung 3.10). Je stärker der Wind ist, desto schneller bewegt sich diese Linie. Zusätzlich kann die Linienanzahl oder Liniendicke der animierten Linie erhöht werden, um verschieden gefährliche Windstärken besser unterscheidbar zu machen. Da dies die Bewegung im Wind symbolisiert, ist auch diese Darstellung sehr intuitiv. Nachteil ist, dass man in einer gleichzeitig laufenden Animation über die Zeit die Tendenz schlechter erkennen kann als bei der Variante, dies über die Pfeilgröße darzustellen.

Die Unsicherheit der Windrichtung lässt sich bei einem Windpfeil auf mehrere Arten darstellen: Eine Möglichkeit ist die Animation 3.3.2, indem man beispielsweise den Pfeil vom Minimum zum Maximum und zurück dreht. Dies kann aber leicht dazu führen, dass der Betrachter fälschlicherweise denkt, der Wind würde sich zu diesem Zeitpunkt immer wieder drehen. Daher ist das Einfügen eines Kreisteiles um den Pfeil [Riv07] eine bessere Darstellung. Dies zeigt dem Benutzer durch einen statischen Pfeil, dass der Wind konstant aus einer Richtung kommt.

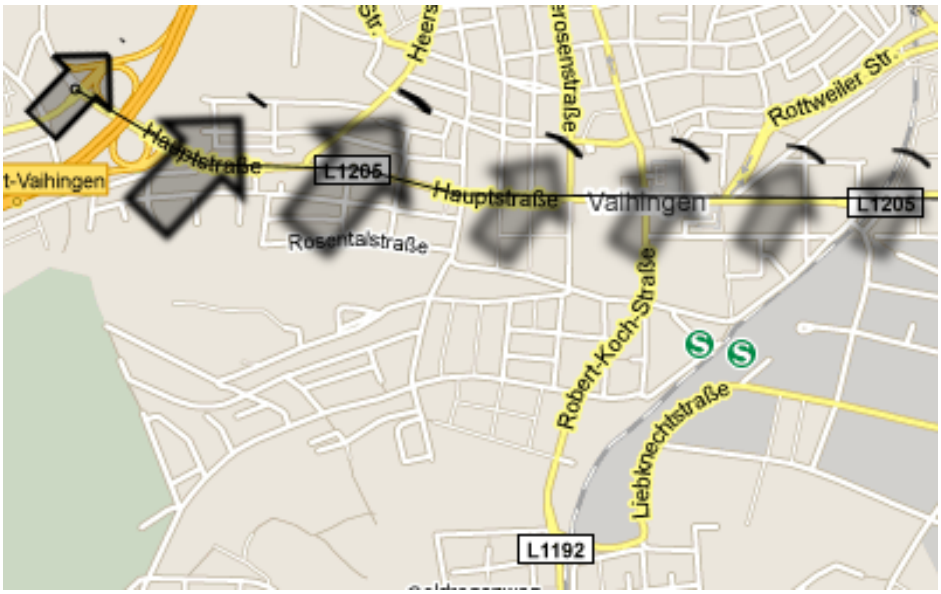

Abbildung 3.9: Darstellung von **Windpfeilen** in der Beispielanwendung. Dabei stellt die Unschärfe die Unsicherheit des Eintretens dar. Der Kreisteil an der Spitze des Pfeils zeigt die mögliche Abweichung der Windrichtung. Je größer der Pfeil hier dargestellt wird, desto stärker ist der Wind.

Abbildung 3.10: Darstellung eines **animierten Windpfeils** in der Beispielanwendung. Dabei wird die Lücke innerhalb des Pfeils umso schneller bewegt, je schneller der Wind ist.

3.3.12 Balken

Eine Möglichkeit, um Werte gut ablesbar darzustellen, sind Balken (Abbildung 3.11). Im Gegensatz zu einem Stackedgraph lassen sich hier auch negative Werte darstellen.

Ein weiterer Vorteil ist die gute Darstellungsmöglichkeit für Unsicherheit. Eine Möglichkeit ist den Balken bis zum Minimum normal darzustellen und von dort bis zum maximalen Wert beispielsweise den Balken heller darzustellen. Eine solche Helligkeitsunterscheidung muss in einer Legende beschrieben werden, um keine Fehlinterpretationen zuzulassen [OM02].

Eine weitere Möglichkeit ist das Darstellen des Minimums und von dort aus ein „T“-Förmigen Glyphen aufzusetzen. Dies lässt sich auch erweitern, so dass der Mittelwert angezeigt wird und das „T“ auch bis zum Minimum seitlich des Balkens angezeigt wird [OM02].

Falls man einen vertikalen Teil der Route hat, muss man, um Überlappungen zu vermeiden, die Balken um 90° drehen. Dies muss immer in die gleiche Richtung geschehen, da man eventuell auch negative Werte darstellen muss und es sonst zu Verwechslungen kommen könnte.

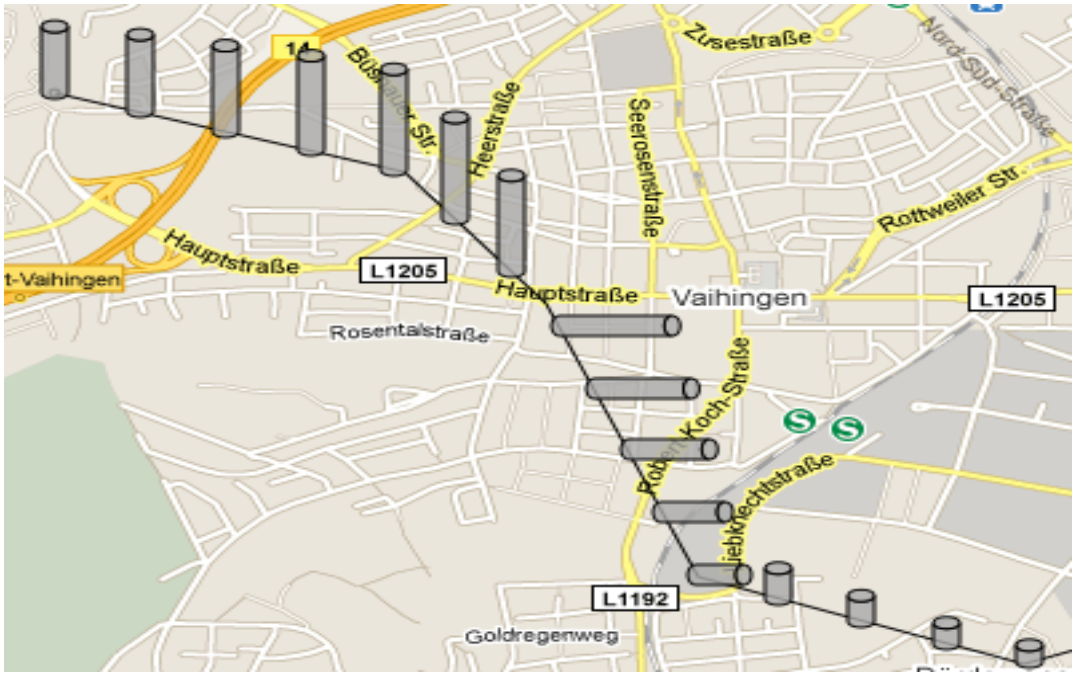

Abbildung 3.11: Darstellung von **Balken** in der Beispielanwendung. Dabei sind bei sehr steilen Strecken die Balken umgekippt, um Überlappungen von hintereinander folgenden Balken zu vermeiden.

3.3.13 Chernoff Gesicht

Ein Chernoff Gesicht bietet die Möglichkeit, viele Werte in einem Glyphen darzustellen. Dazu wird ein Gesicht gemalt, das verschiedene Merkmale hat. Diese lassen sich dann je nach dargestelltem Wert größer, breiter oder anderweitig verändert darstellen [FR81]. Beispielsweise lässt sich mit den Ohren die Lärmbelastung darstellen. Je kleiner die Ohren, desto weniger Lärm. Die Nase und Augen lassen sich gut für die Pollenbelastung verwenden. Pupillen könnten die Tageshelligkeit anzeigen und die Mundwinkel die allgemeine Luftqualität.

Die Unsicherheit lässt sich durch die in Linien 3.3.2 beschriebenen Methoden oder durch Animation 3.3.2 darstellen.

4 Implementierung

4.1 Eingesetzte Technologien

Da die Software für möglichst viele Benutzern zugänglich sein soll, wurde eine nativ browserbasierte Darstellung gewählt. Dies hat den Vorteil, dass grundsätzlich mit allen HTML5 fähigen Browsern die Darstellungen funktionieren, ohne dass etwas zusätzlich installiert werden muss.

Daher bietet sich das Google Web Toolkit[1] (GWT) an. Mit diesem ist es möglich, Java-Code in JavaScript zu übersetzen. Da man Java Programme, im Vergleich zu JavaScript, viel besser warten kann und man einfacher komplexe Software erstellen kann, wurde sich für dieses Framework entschieden.

Als Kartendienst wurde Google-Maps[2] ausgewählt. Dieses bietet eine sehr gute und erfolgreiche API [Hom07] und viele zusätzlicher Funktionen wie ein Verkehrsoverlay oder Streetview.

Für die Anzeige der einzelnen Darstellungen gibt es grundsätzlich zwei Möglichkeiten in HTML5. Die Erste ist der 2D-Canvas, auf dem frei und rasterisiert gezeichnet werden kann. Dieser wird unter anderem von Protovis[3] verwendet. Protovis ist ein JavaScript Framework und implementiert ein sehr weites Spektrum an Diagrammen. Ein GWT-Wrapper für Protovis ist im Choosel-Projekt[4] enthalten.

Die zweite Möglichkeit ist die Verwendung von SVG (Scalable Vector Graphics)[5]. Dies ist ein leichtgewichtiger Standart, mit dem man gut Anwendungen bauen kann, die auf Grafik ausgerichtet sind [Qui03]. Diese werden als DOM Elemente ebenfalls von HTML5 kompatiblen Browsern unterstützt. SVG besteht aus einzelnen Elementen, die in Gruppen zusammengefasst und dann transformiert werden können. Frameworks, die diese Möglichkeit nutzen sind zum Beispiel Raphael[6] oder lib-gwt-svg[7].

[1] http://code.google.com/intl/de-DE/webtoolkit/
[2] http://code.google.com/intl/de/apis/maps
[3] http://vis.stanford.edu/protovis
[4] http://code.google.com/p/choosel/wiki/ProtovisGWT
[5] http://www.w3.org/TR/2011/WD-SVG11-20110512
[6] http://raphaeljs.com
[7] http://code.google.com/p/lib-gwt-svg/

Als Injection-Framework 2.5 wurde Guice, bzw. die GWT-Implementierung Gin[8] verwendet. Dies bietet eine sehr einfache API und Konfiguration. Zudem hat es so gut wie keinen Einfluss auf die Performance des Endprodukts und erhöht die Wartbarkeit und Konfigurierbarkeit des Frameworks.

Es wurde zudem ein Client ohne Serveranbindung erstellt. Die Serverseite kann also komplett vom Verwender des Frameworks nach belieben gewählt werden. Als Datenquellen wurden Dummydatenquellen verwendet, die die Daten mit vordefinierten Algorithmen erstellen.

4.1.1 Evaluierung und Technologieentscheidungen

Protovis deckte zur Zeit der Diplomarbeit leider nicht alle benötigten Funktionalitäten ab, lies sich aber dementsprechend erweitern. Es bietet zwar vielerlei Diagramme, dennoch wurde es lediglich für die Anzeige der Heatmap verwendet, da SVG weit aus mehr Möglichkeiten bot.

Bei den Versuchen mit Raphael zeigten sich schnell dessen Grenzen. Es implementiert Gruppen von Elementen und deren Transformationen selbst, statt die in SVG vorgesehenen Elemente zu verwenden. Es traten bei diesen Transformationen einige Bugs auf, die dem Konzept von Raphael geschuldet waren. Zudem funktioniert kein mehrmaliges Transformieren, was für eine gute abstrakte Implementierung nötig ist. Zudem war die Performance im Gegensatz zu der nativen SVG-Lösung relativ schlecht. Aus diesen Gründen wurde die native Lösung gewählt. Lib-gwt-svg ist, im Gegensatz dazu, eine reine GWT-Implementierung und bietet Zugriff auf die SVG-DOM Elemente. Durch die vielfach höhere Performance und den besseren Möglichkeiten wurde dieses Framework daher Raphael vorgezogen.

4.2 Architektur

In diesem Kapitel werden zunächst die grundlegenden Architekturentscheidungen vorgestellt. Danach werden die konkreten Darstellungen und die dort eingesetzten Algorithmen und Ideen vorgestellt.

4.2.1 Erweiterbarkeit

Da die Software später ins PESCaDO-System eingebaut und erweitert werden soll, ist es wichtig, dass das Framework gut erweiterbar und konfigurierbar ist. Dies wird, neben der Klassenstruktur (Abbildung 4.1), durch Guice bzw. Gin ermöglicht. Es wurde darauf geachtet, einzelne Bereiche und Algorithmen damit ersetzen zu können. Zudem wurde die Software in verschiedene Klassen aufgeteilt, die jeweils einen Aufgabenbereich erfüllen:

[8] http://code.google.com/p/google-gin/

- Data -> Um was für Daten handelt es sich, Hierarchie Information.
- DataSource -> Beschaffen der Daten und anderer relevanter Informationen.
- DataInfo -> Informationen über die von der DataSource gelieferten Daten und deren Anwender.
- DataViewOverlay -> Für Darstellungen direkt auf der Karte.
- DateViewIcon -> Darstellung für ein Glyph 3.3.8.
- IconRenderer -> Anordnung der Glyphen als Flächen, Route oder andere Darstellungen.

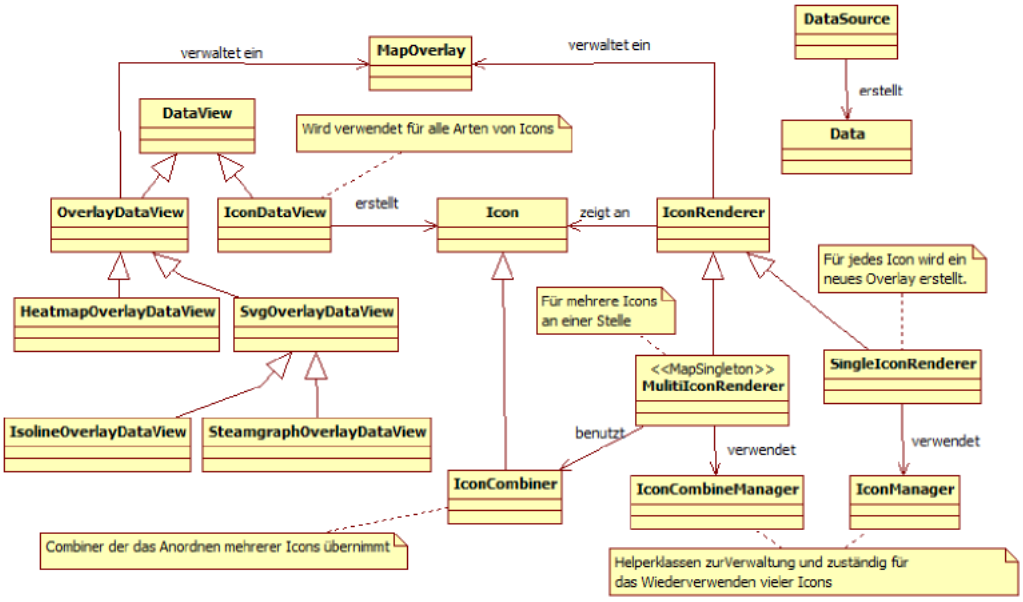

Abbildung 4.1: Klassendiagramm des Frameworks mit einigen Beispielklassen.

4.2.2 Daten und deren Quellen

Das Modell von Daten besteht aus einer Oberklasse „Data“. Darunter gibt es zwei Datenarten: UncertainData für Daten, die Werte mit Unsicherheit beinhalten und Complexdata für zusammengesetzten Daten. Complexdata wird verwendet, wenn in einer Darstellung mehrere Daten benötigt werden, beispielsweise Winddata, welche aus Windstärke und Windrichtung zusammengesetzt ist. Viele Datenklassen der UncertainData lassen sich in IntervalData zusammenfassen. Hat man mehrere Intervaldaten, so kann man die MultiIntervalComplexData verwenden. Dies ist beispielsweise bei Luftqualitätsdaten der Fall.

Um an diese Daten heranzukommen, werden asynchrone DataSources verwendet. Die Darstellungen bekommen eine DataSource bei ihrer Initialisierung, über die sie die von ihnen

benötigten Daten mit Orts und Zeitangabe abfragen können. Um die Kommunikation zum Server später unterstützen zu können, wird die Antwort asynchron 2.4 dem Anfragenden zurückgegeben.

DatenSources bieten zudem Zugriff auf sogenannte DataInfos. Diese bieten Informationen über benutzerspezifische Interpretationen der Daten. Wenn ein Benutzer beispielsweise auf bestimmte Pollen allergisch ist, so werden die Ampel-Schwellenwerte hier angepasst. Diese werden dann von den Darstellungen verwendet, um die Daten möglichst intuitiv und für den Benutzer angepasst darzustellen.

4.2.3 Unsicherheit

Die Unsicherheit wird in diesem Projekt direkt im Datenmodell berücksichtigt. Dazu speichert UncertainData nicht den konkreten Wert, sondern die Bereichsunsicherheit, sprich Minimum und Maximum des zu erwartenden Wertes. Unterstützt eine Darstellung nur einen konkreten Wert, so wird der Mittelwert verwendet.

Die Eintrittswahrscheinlichkeit wird in einem separaten Wert, auch innerhalb von UncertainData als Attribut, gespeichert.

Liegen zu einem angefragten Ort/Zeit keine Daten vor, so übernimmt die DataSource oder die von ihr verwendeten Services die Interpolation. Dabei kann auch der Bereich oder die Wahrscheinlichkeit angepasst werden. Diese Berechnungen sind aber nicht Teil dieser Arbeit und werden später vom Framework unterstützt.

4.2.4 CurrentData

Um überall Zugriff auf die aktuelle Position, wie in Kapitel 3.3.1 beschrieben, erhalten zu können, wird bei den Initialisierungen der Objekte immer ein CurrentData Objekt übergeben bzw. per Injection 2.5. Dieses enthält neben der Route mit Orts- und Zeitkoordinaten auch die aktuell anzuzeigende Position. Mittels des Listener Patterns[9] lassen sich bei Änderungen die Darstellungen an die neuen Daten anpassen.

Wie in Kapitel 3.3.2 beschrieben, ist es zudem sinnvoll, eine einheitliche Größe für alle Darstellungen zu verwenden. Diese Zoomstufe ist ebenfalls im CurrentData Objekt abfragbar.

4.2.5 Allgemeine Widgets

Um das CurrentData Objekt vom Benutzer änderbar machen zu können, können beispielsweise wie in Abbildung 4.2 Schieberegler verwendet werden, die auf die Karte als Controls eingefügt werden. Neben dem aktuellen Ort kann so auch der aktuelle Zeitpunkt oder

[9] http://de.wikipedia.org/w/index.php?title=Observer_(Entwurfsmuster)&stableid=86993850

die Zoomstufe der Darstellungen angepasst werden. Auch lässt sich auf diese Weise eine einschaltbare Animation der Daten über die Zeit realisieren.

Abbildung 4.2: Schieberegler für Änderungen des CurrentData-Objekts. Dabei können Zeit oder Ort (einstellbar in der Combobox) mit dem Schieberegler angepasst werden. Es besteht die Möglichkeit, den Schieberegler animiert über den „Play"-Button durchlaufen zu lassen. Zudem gibt es die Möglichkeit, den aktuellen Punkt an die Stelle der Route setzen zu lassen, der am nächsten an der Mausposition ist. Darunter gibt es einen Schieberegler für die Zoomstufe der Widgets 3.3.2.

Zusätzlich kann ein Overlay eingefügt werden, auf dem die Route dargestellt wird. Eine Darstellung für den aktuellen Punkt kann so, wie in Abbildung 4.3 gezeigt, eingefügt werden. Ebenfalls können beispielsweise über Marker die einzelnen Routenpunkte explizit dargestellt und über Drag&Drop angepasst werden.

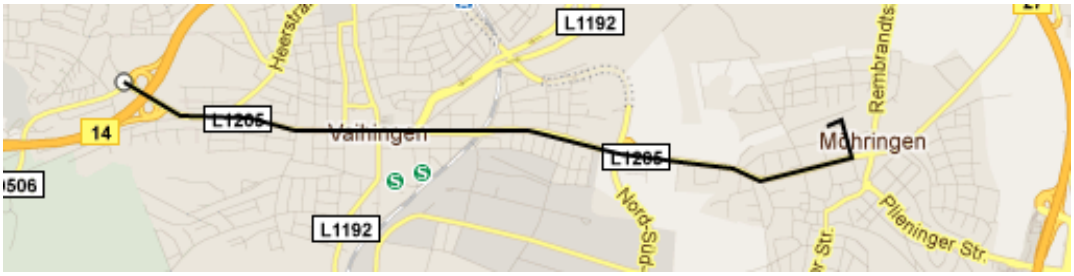

Abbildung 4.3: Angezeigte Route und des aktuellen Punktes auf der Karte.

Um diese Darstellungen aktuell zu halten, registrieren diese ChangeListener, welche aufgerufen werden, wenn sich etwas daran ändert, auf dem CurrentData-Objekt. Falls beispielsweise der Schieberegler den aktuellen Ort des Punktes ändert, so bekommt dies das CurrentPoint-Overlay, welches den aktuellen Punkt auf die Karte malt, dies über diesen Listener mit und kann den angezeigten Kreis auf der Karte an die neue Position verschieben.

4.3 Anwendungsszenarien

In diesem Kapitel werden Anwendungsszenarien beschrieben, auf die diese Architektur abgestimmt sein soll und es werden Beispiele für deren mögliche Umsetzungen gegeben.

4.3.1 Ich will morgen mit dem Fahrrad zur Arbeit fahren.

In diesem Szenario ist ein fester Start und ein festes Ziel vorgegeben. Es geht dabei vor allem um den Zeitpunkt des Losfahrens, um optimale Bedingungen auf der Fahrt vorzufinden.

Es ist eine feste Route vorgegeben, bei welcher der Startzeitpunkt variiert werden kann. Um dies zu realisieren, kann man bei Verschiebung des Startzeitpunktes alle Zeitpunkte in CurrentData dementsprechend anpassen. Es lässt sich also alles über die alleinige Manipulation des CurrentData-Objektes implementieren.

4.3.2 Ich will morgen Tennis spielen.

Man will in diesem Szenario wissen, ob man draussen Tennis spielen können oder ob man in der Halle einen Platz reservieren muss. Zusätzlich will man den Weg zum Tennisplatz mit dem Fahrrad zurücklegen, falls das Wetter es erlaubt.

Abweichend zum ersten Szenario 4.3.1 hat man hier den Fall, dass man sich für eine gewisse Zeit an einer Stelle aufhält, bevor man sich weiter auf der Route bewegt. Dies lässt sich mittels eines weiteren Routenpunktes mit gleichem Ort, aber anderem Zeitpunkt, am Tennisplatz realisieren. Dabei ist zu beachten, dass sich Routendarstellungen 3.3.1 für dieses Szenario eventuell nicht eignen, da für einen Ortspunkt auf der Route nur ein dargestellter Wert vorgesehen ist.

4.3.3 Ich will eine Radtour in einem Gebiet machen.

Der Unterschied zu den bisherigen Szenarien liegt darin, dass keine bestimmte Route vorgegeben ist, sondern diese erst geplant werden soll.

Wie in Kapitel 4.2.5 beschrieben, können Marker in die Karte eingefügt werden, die verschiebbar sind. Beim Verschieben eines Markers werden die entsprechenden Routenpunkte in CurrentData aktualisiert und so die Darstellungen an die neue Route angepasst. Zudem können neue Punkte eingefügt oder vorhandene entfernt werden, um die Route genau einstellen zu können.

4.3.4 Eine Gruppe Rentner will eine Wandertour machen.

Da Menschen unterschiedlich empfindlich auf Umweltfaktoren sein können, ist es wichtig, dies zu berücksichtigen. Will beispielsweise eine Rentnergruppe einen Ausflug planen, so müssen wahrscheinlich die Grenzwerte für die Ampelwerte 3.1.1 an deren Bedürfnisse angepasst werden.

Dies kann mittels der DataInfos 4.2.2 geschehen, die von den DataSources 4.2.2 bereitgestellt werden. Über diese werden die Grenzwerte an die Darstellungen übergeben, die zum Beispiel bei den Ampeldarstellungen verwendet werden.

4.3.5 Ich will in einem Gebiet das Wetter wissen.

Hier hat man im Gegensatz zu den anderen Szenarien den Fall, dass es keine Route, sondern ein Gebiet als Ausgangspunkt gibt.

Um dies mit dem Framework zu realisieren, werden zwei Ortspunkte benötigt, die in zwei gegenüberliegenden Ecken eines rechteckigen Gebietes liegen. Man stellt hierbei den aktuellen Punkt und die Route nicht dar. Zudem wird der hierbei überflüssige Orts-Schieberegler ausgeblendet. Man kann bei dieser Methode auch nur Flächendarstellungen 3.3.1 verwenden, da dies die einzige Darstellungsart für Wetterdaten ist, die sich nicht auf den aktuellen Ort bezieht.

Um die Uhrzeit einzustellen, können die üblichen Widgets 4.2.5 verwendet werden. Dafür muss lediglich bei dem ersten der zwei Punkte der Startzeitpunkt, und beim Anderen der Endzeitpunkt des darzustellenden Zeitbereiches gesetzt werden.

4.4 DataViews

4.4.1 OverlayDataViews

Canvasarten

Die SVG-Elemente müssen nun auf der Karte platzieren werden. Fügt man den Canvas direkt beim zu zeichnenden Element ein, bekommt man Probleme mit bewegten Objekten, die sich aus diesem Bereich herausbewegen. Es gibt mehrere Möglichkeiten dieses Problem zu lösen:

1. Dynamischer Canvas: Es wird immer die linke obere und die rechte untere Ecke der darzustellenden Grafik auf der Karte berechnet und der Canvas wird dann dahin geschoben und vergrößert.

Der Vorteil dieses Konzeptes ist, dass man immer einen sehr kleinen Canvas hat. Dafür muss bei jedem Rendern die Größe und Position neu berechnet werden und die Elemente darin dann richtig verschoben werden. Dies müsste über eine Zwischenschicht gelöst werden, da sich auch die Pixelpositionen ständig ändern würden.

2. Fester Canvas, der sich nicht über die Maps-API mitgescrollt sondern sich intern selbst beim scrollen der Karte verändert.

Man legt den Canvas, unabhängig von GoogleMaps, über die Karte. Um das Scrollen zu ermöglichen, fängt man über die Maps-API deren Scrollevents ab, fragt die Maps-API nach dem aktuell dargestellten Kartenausschnitt und transformiert dementsprechend den Inhalt des Canvas.

Man hat dabei die volle Kontrolle darüber was angezeigt wird und hat die Probleme mit sich verändernden Elementen, die man beim Dynamischen Canvas hatte, nicht mehr.

Bei diesem Ansatz umgeht man allerdings die stabile und hochperformante Maps-API und müsste vieles, was durch diese abgedeckt ist, selbst implementieren.

3. Fester Canvas, der mittels Map-API scrollt und nur verschoben wird, sobald in Bereiche gescrollt wird, die nicht von dem Canvas abgedeckt sind.

Man kann nun die Vorteile die Methoden 1 und 2 kombinieren, indem man einen Canvas mit fester Größe auf der Karte platziert. Dabei verwendet man beispielsweise einen Canvas, der doppelt so groß ist, wie der darzustellende Bereich. Man verschiebt den Canvas und die darin enthaltenen Elemente nur, wenn man aus dem Canvasbereich heraus scrollt. Hier verwendet man zwar das Scrollen der Maps-API, jedoch muss man immer noch mit Hilfe der Scrollevents von GoogleMaps herausfinden, wann man den Canvas verschieben und die enthaltenen Elemente richtig transformieren muss.

4. Riesen Canvas. Es wird ein Canvas mit Millionen von Pixeln Größe auf die Karte gelegt.

Mittels der Maps-API lassen sich Geo-Koordinaten in Pixel umwandeln. Zu beachten ist, dass diese Koordinaten auch negativ sein können. Man erstellt bei dieser Methode einen Canvas, der an einer Stelle platziert wird, die immer außerhalb des Bereiches liegt wie zum Beispiel $(-100000000/-100000000)$ und einer maximalen Größe. Man transformiert diesen, so dass der $(0/0)$-Punkt mit dem $(0/0)$-Punkt der Maps-API übereinstimmt. Somit kann man ohne Umrechnungen direkt auf den Canvas mit den Kartenkoordinaten zeichnen.

Da diese Implementierung sehr simpel ist, ist diese auch am wenigsten anfällig für Fehler. Nach Tests spielt diese Größe für die heutigen Browser keine große Rolle.

Entscheidung

Durch die Paradigmen „Don't suboptimzie“ und „Keep it simple“ wurde die vierte Variante ausgewählt. Falls die Performance einbricht, kann auch auf die dritte Variante umgestellt werden. Dies ist nachträglich noch gut umstellbar, da beide jeweils eine SVG Gruppe verwenden können, in welche die Darstellungen ihre SVG-Element einfügen.

Ein Problem gab es bei der vierten Variante jedoch. Bei der zur Arbeit verwendeten Chrome Version wurde durch JavaScript die Integerzahl der Canvasgröße auf width=„1e+008px“ gesetzt, mit dem Chromes HTML-Engine nicht zurecht kam. Dieses Problem hat sich jedoch mittels Verwenden von Strings beim Setzen des Attributes umgehen lassen. Innerhalb der SVG-Elemente funktionierte diese Darstellung bereits.

DataViewOverlay-API

Die API der DataViewOverlays ist nach dem Vorbild der Maps-API so erstellt, dass grundsätzlich kein SVG oder HTML5-Canvas verwendet werden muss. Es sind also auch beliebige andere Konzepte damit realisierbar.

Diese Overlays sind generisch und geben über diesen Typ die Art der Daten an, die sie anzeigen können. Beispielsweise können Heatmaps alle Arten von Intervaldaten anzeigen

und erben somit von DataViewOverlay<IntervalData>. Dies definiert den Typ der DataSource, welche die DataView erhält.

Zudem kann von der DataView ein Options-Objekt von aussen abgefragt werden. Hiermit kann die DataView eingestellt werden. Dies kann später durch Algorithmen dann auf die jeweiligen Anwendungsfälle eingestellt werden.

Weitere Einstellungen können von der DataSource mittels der DataInfos an die DataView übergeben werden. Dies sind beispielsweise Grenzwerte oder Einheiten, die auf den Anwender angepasst sind.

Da viele Overlays Legenden verwenden, können diese auch aus der DataView geholt und dann zentral angezeigt werden.

4.4.2 IconDataViews

Wie in Kapitel 3.3.8 beschrieben, lassen sich Icons in vielerlei Darstellungsarten 3.3.1 einsetzen. Um dies nicht bei jedem Icon neu bauen zu müssen, wurde das IconDataView Konzept eingeführt. Diese Klasse verwaltet nur ein einzelnes Icon. Die Positionierung und das Einstellen des Wertes übernimmt ein sogenannter IconRenderer.

Es gibt mehrere IconRenderer, die für die einzelnen Darstellungsarten verwendet werden:

Flächen-Renderer

Dieser erstellt ein Raster auf der anzuzeigenden Fläche, abhängig von der Maximalen Größe der Icons 4.2.4.

Route-Renderer

Der Route-Renderer soll die Glyphen möglichst nahe zusammen, aber ohne Überlappungen, auf der Route darstellen. Dazu wird folgender Algorithmus verwendet:

Man startet am ersten Punkt der Route und zeichnet den ersten Glyph. Danach berechnet man die Ausmaße des Glyphen. Unter der Annahme, dass der nächste zu zeichnende Glyph ungefähr genauso groß sein wird, wird diese berechnete Breite verdoppelt und ein Randwert addiert. Man erhält ein Rechteck um den gerade gezeichneten Glyphen, auf dessen Kanten der Mittelpunkt des nächsten Glyphen liegen kann. Man muss als nächstes den Schnittpunkt der Route mit diesem Rechteck berechnen. Als Route wird hierbei nur die Route von dem zuletzt gezeichneten Glyphen verwendet, um einen eindeutigen Schnittpunkt zu erhalten. Dies wird nun über die komplette Route iteriert (Abbildung 4.4).

Eine weitere Voraussetzung für diesen Algorithmus ist, dass man den Glyphen genau mittig auf dem Punkt platziert.

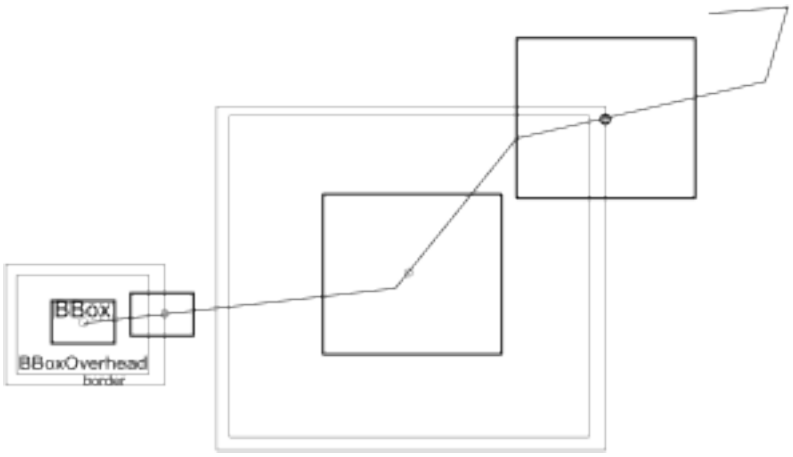

Abbildung 4.4: Routenalgorithmus für Glyphen: Es sind hier zwei Beispiele für die Anwendung des Algorithmus zu sehen.

Punkt-Renderer

Dieser zeichnet das Icon an die aktuell darzustellende Position 4.2.4.

Separat-Renderer

Wie in Kapitel 3.3.1 beschrieben erstellt dieser auf oder neben der Karte einen Bereich, in dem das Icon, mit den aktuellen-Punkt 4.2.4 dargestellt wird.

Kombinierte-Renderer

Oft besteht der Anwendungsfall, mehrere Icons gleichzeitig in der selben Darstellungsart zu zeichnen. Dazu gibt es jeden Renderer auch als Kombinierter Renderer. Dieser verwendet einen Algorithmus, um mehrere Icons um den Punkt herum anzuordnen. Dies ist nur ansatzweise implementiert, da dies nicht Teil dieser Diplomarbeit ist.

4.4.3 Heatmap

Die Heatmap ist mit Hilfe von Protovis implementiert. Nachdem die in einem Raster angeordneten Daten von der DataSource bei der DataView angekommen sind, werden diese Werte mit Hilfe einer linearen Skalierung (linear Scale in Protovis) auf die Farben abgebildet. Innerhalb der Image-Funktion, bei der die Pixelfarben bestimmt werden, können nun auch die Farbänderungen, abhängig vom Unsicherheitsfaktor, durchgeführt werden.

Um die zu verwendenden Farben zu bestimmen, wird zunächst in den DataInfos nach konkreten Farbwerten geschaut. Sind diese nicht vorhanden, so werden Ampelwerte verwendet.

Sind auch diese nicht vorhanden, wird das Maximum und Minimum der angekommenen Daten berechnet und diese als Schwellwerte herangezogen.

Um die Farben zwischen den Datenpunkten zu ermitteln, gibt es die Möglichkeit des Pre- oder Postshading. Beim Postshading werden zunächst die Datenwerte der einzelnen Bildpunkte berechnet und diese dann jeweils in Farben umgerechnet. Beim Preshading dagegen, werden nur die Datenpunkte in Farben umgerechnet. Für die Interpolation werden hier dann die Farbwerte verwendet.

Um die Implementierung möglichst einfach und effizient zu realisieren, wurde das Preshading folgendermaßen verwendet: Man baut mittels Protovis ein Bild auf, dessen Pixelzahl der der Datenmatrix entspricht, die man ausgelesen hat. Die Umrechnung von Werten auf deren konkrete Farben wird über einen linearen PVScale von Protovis gemacht. Dieser wird mit den Farben und den Grenzwerten initialisiert und kann uns dann die Farbe für einen bestimmten Wert ausrechnen. Danach streckt man das Bild mittels CSS Attributen auf die Größe, die man für die Karte benötigt. Das Interpolieren der Farben zwischen den Datenpunkten überlässt man so den hochperformanten Rendering Engines des Browsers.

Für die Legende kann man die Farben an bestimmten Grenzwerten mit den dazugehörigen Werten darstellen. Es hat sich aber schnell gezeigt, dass man dabei die Werte auf der Karte nicht gut zuordnen kann. Daher wurde die Legende in einen Farbverlauf umgewandelt. Dies wurde, ähnlich wie bei der Heatmap selbst, mit Protovis realisiert. Es wurde ein Minimal-, und ein Maximalwert verwendet und diesen in N gleich große Bereiche eingeteilt. Danach wurden diese verwendet um ein 1 Pixel breites und N Pixel hohes Bild eingetragen und eingefärbt. Dies wurde dann wieder gestreckt, so dass der Farbverlauf entsteht.

4.4.4 Isolinien

Isolinien können sowohl nativ mit einem Canvas als auch mit SVG Elementen implementiert werden, da die Algorithmen die Kanten der Bereiche als Ergebnis haben.

Der Algorithmus, der für die Berechnung verwendet wurde, ist dem Algorithmus aus [LP09] nachempfunden. Dabei wird als Ausgangspunkt wieder eine Datenmatrix verwendet. Jede Linie der kleinsten Rechtecke dieser Matrix wird nun auf Bereichsüberschreitungen untersucht. Wie in [LP09] bewiesen, sind entweder 0, 2 oder 4 Bereichsüberschreitungen pro Bereich möglich. Zu beachten ist hier, dass in einem Feld auch mehrere Bereiche überschritten werden können.

Um Möglichkeiten von SVG ausnutzen zu können, wird zusätzlich ein Pfad der zusammengehörigen Bereichslinien berechnet. Dies bietet zum Beispiel die Möglichkeit einer besseren Abrundung der Ecken, die durch diesen Algorithmus zwangsläufig entstehen.

4.4.5 Stackedgraph

Um den Stackedgraphen zu implementieren, benötigt man zunächst einen Algorithmus, mit dem man die Pfade für die Linien parallel zur Route berechnen kann. Dieser wird in Kapitel 4.4.7 später erläutert. Da für einen Stackedgraphen Flächen benötigt werden, muss dieser Algorithmus dann noch leicht modifiziert werden, so dass er auch den umgedrehten Pfad, also vom Ende zum Anfang der Route, berechnen kann. Um nun den Pfad für eine Fläche zu bekommen, kann man diese Pfade hintereinander anfügen und erhalten den Pfad für die Fläche, die man dann einfärben kann.

Um die Werte übereinander gestapelt anzuzeigen, wie es beim Stackedgraphen nötig ist, addiert man bei den Werten jeweils alle zuvor verwendeten Werte an den jeweiligen Stellen, sprich die darunter liegenden Werte.

4.4.6 Liniendiagramm

Für ein Liniendiagramm wird, wie beim Stackedgraph, der Algorithmus für die Linien, die parallel von Routen verlaufen, benötigt. Man kann die vom Algorithmus gelieferten Pfade direkt in das Diagramm zeichnen. Um Unsicherheit abzubilden, werden wie beim Stackedgraphen zwei Pfade zu einer Fläche zusammengesetzt, in dem man den minimalen und den maximal möglichen Wert als den oberen und unteren Pfad verwendet. Dabei wird wieder einer der beiden Pfade rückwärts berechnet, um später die komplette Fläche mit einem Pfad erstellen zu können.

Damit der Benutzer die Linien besser interpretieren kann, wurde der Hintergrund des Diagramms noch eingefärbt. Dies wurde ebenfalls mit dem in Stackedgraph 4.4.5 beschriebenen Algorithmus gemacht, indem die festen Index-Grenzwerte für jeden Punkt verwendet wurden. Da der Hintergrund sehr transparent gehalten wurde, um möglichst wenig Störungen mit anderen Darstellungen zu riskieren (Kapitel 2.1) wurden noch zwei Grenzlinien in den entsprechenden Farben eingefügt.

4.4.7 Darstellungen von Linien an einer Route

Bei der Darstellung von Linien an einer Route sind vor allem die Eckpunkte, an denen die Route abknickt, zu beachten.

Im Falle eines **Innenwinkels** können die eingehenden Linien einfach an deren Schnittpunkt abgeschnitten werden. Dies ist in Abbildung 4.5 veranschaulicht.

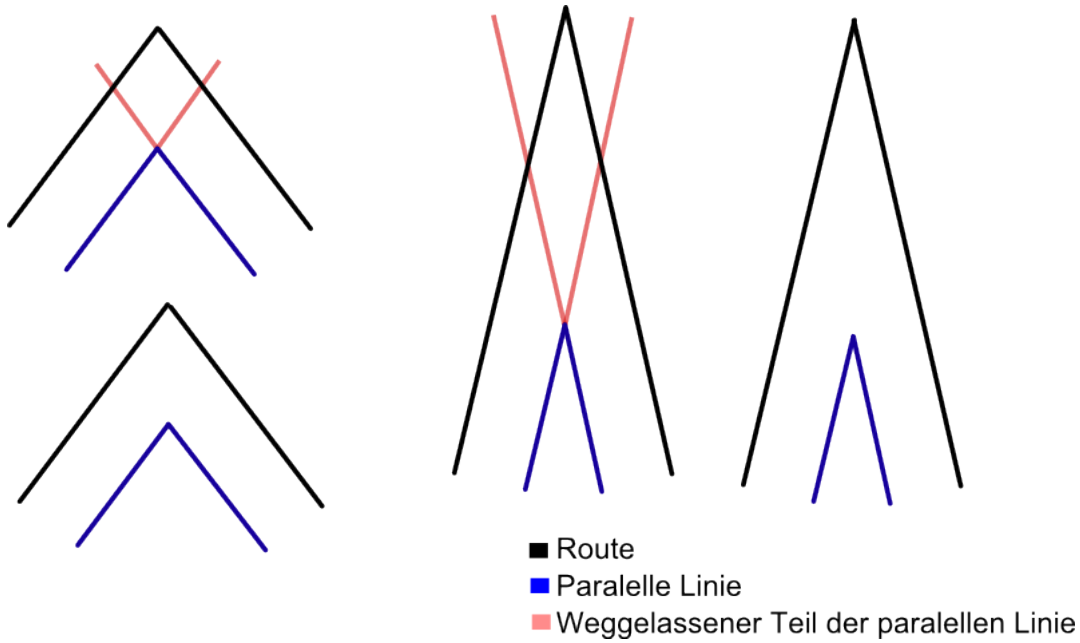

Abbildung 4.5: Verbinden von parallel laufenden Linien mit Innenwinkel durch Abschneiden der Linien am Schnittpunkt.

Handelt es sich jedoch um einen **Außenwinkel**, so gibt es die Möglichkeit die dadurch auftretende Lücke mit geraden, spitzen, abgerundeten oder kreisrunden Teilen auszufüllen. Die Linien gerade zu verbinden, ist die einfachste Möglichkeit, die kaum Aufwand benötigt. Verbindet man die beiden Punkte nicht gerade sondern über einen Kreis, der seinen Mittelpunkt an dem Punkt auf der Route hat, so erhält man die Kreisdarstellung. Für die Abgerundeten- und Spitzendarstellung werden die Linien verlängert und deren Schnittpunkt für die Darstellung verwendet. Dabei wird bei der Spitzen-Darstellung gerade Linien verwendet. Um die Abgerundete Darstellung zu bekommen wird eine Bézier Kurve gezeichnet und die Steigungspunkte auf diesen Schnittpunkt gesetzt.

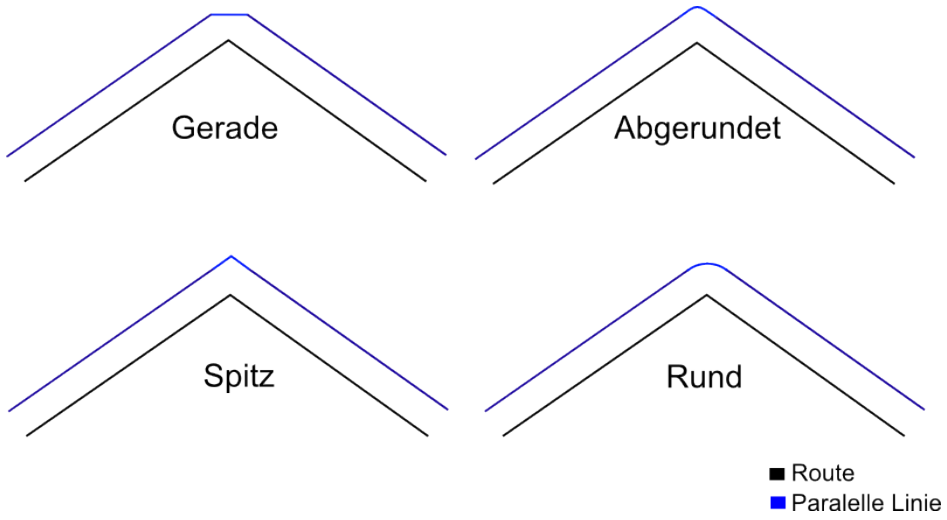

Abbildung 4.6: Verbinden von parallel laufenden Linien mit akzeptablem Winkel.

Die in Abbildung 4.6 aufgeführten Varianten scheinen alle annehmbar zu sein, dies ändert sich aber schnell, wenn sich der Winkel noch weiter vergrößert.

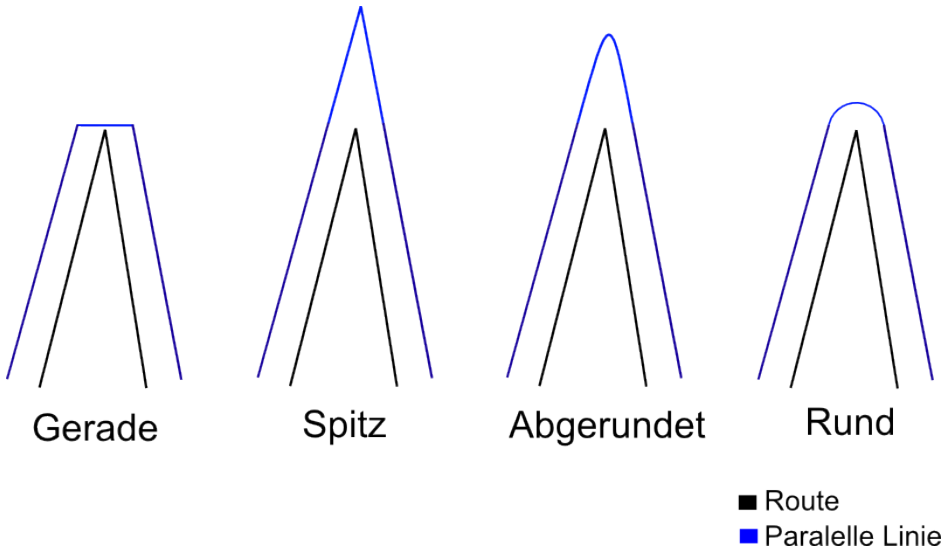

Abbildung 4.7: Verbinden von parallel laufenden Linien mit großem Winkel.

Man sieht in Abbildung 4.7 sofort, dass abgerundet und spitze Verbindungen für diese großen Winkel nicht geeignet sind. Je größer der Winkel wird, desto länger sind diese Verbindungen. Dies führt bei 360° Winkeln zu unendlich großen Verbindungen. Aber auch die gerade Verbindung verfälscht hier das Diagramm stark. Die beste Möglichkeit bei großen Winkeln sind also kreisrunde Verbindungen.

5 Evaluation

5.1 Einführung

Am Ende dieser Arbeit wurde eine Evaluation durchgeführt. Diese wurde in Form einer Demo-Anwendung realisiert, welche zunächst in die Software einführt, dann die einzelnen Darstellungen vorstellt und am Ende einige Kombinationen von Darstellungen zeigt. Zudem werden vielerlei zusätzliche Informationen erfragt, wie die benutzte Hardware oder den Beruf. Um festzustellen, ob eventuelle Probleme mit den Darstellungen durch Farbfehlsichtigkeit 2.1.2 entstanden sind, wurde hierfür auch ein entsprechender Test eingebaut.

In diesem Kapitel wird nun Schritt für Schritt der Evaluation vorgestellt und jeweils die entsprechenden Resultate dargestellt. Am Ende wird dann ein Fazit der ganzen Evaluation gezogen.

5.2 Implementierung und Tutorial

Um diese Evaluation zu realisieren wurde, mit Hilfe des zuvor erstellten Frameworks, eine Software erstellt. Zusätzlich wurde für das Tutorial dieses Framework minimal erweitert, so dass einzelne Schieberegler (Abbildung 4.2) ausgeblendet werden können. Zudem besteht die Möglichkeit in den Evaluationsschritten die Karte und die angezeigten Darstellungen anzupassen. Die Eingabemöglichkeiten, wie beispielsweise die gestellten Fragen und Kommentarmöglichkeiten, werden in einem GoogleMaps-Control unten rechts angezeigt. Um die komplette Karte sehen zu können lässt sich dieses Fenster noch ausblenden (Abbildung 5.1).

Unter jeder Darstellung gab es die Möglichkeit über ein Freitextfeld einen Kommentar zu der Darstellung oder dem Test zu hinterlassen. Dies ist sehr vielen Benutzern verwendet worden.

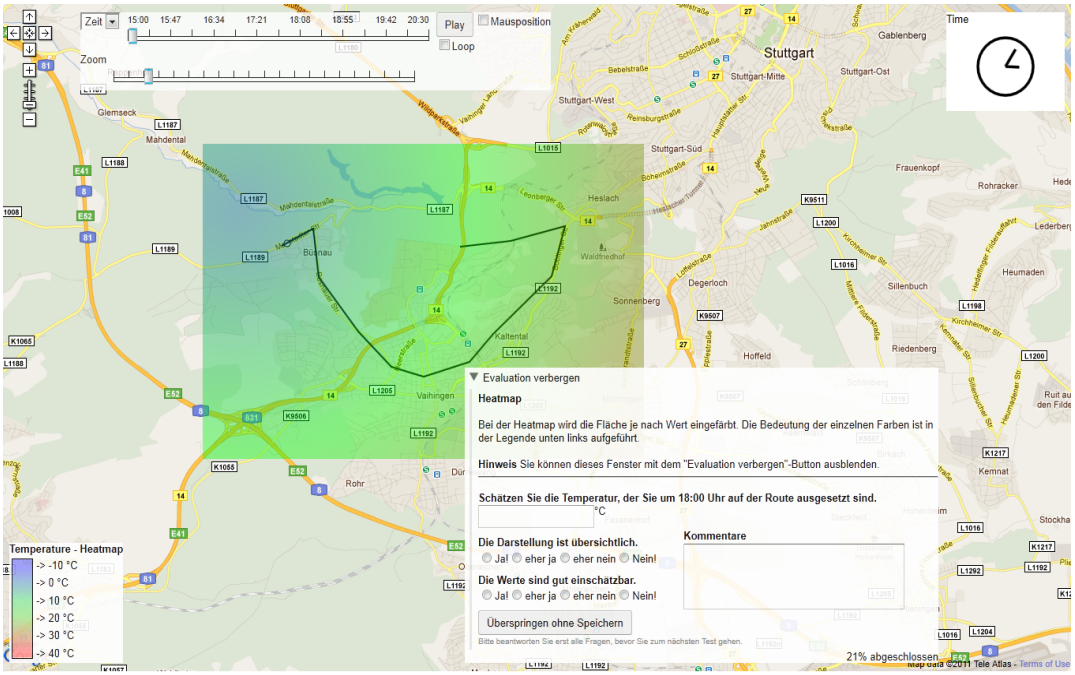

Abbildung 5.1: Ausschnitt der Evaluation, in dem das **Evaluationsfenster** sichtbar ist.

Die verwendeten Daten wurden für jeden Benutzer mit den gleichen Algorithmen erstellt. Zudem wurden mehrere unterschiedliche Routen verwendet, um den Lerneffekt der Benutzer, im Laufe der Evaluation, zu verringern.

5.3 Benutzer

Insgesamt haben **91 Benutzer** am Test teilgenommen. Davon haben **55 Benutzer** die Evaluation bis zum Ende durchgeführt. Der Großteil der Benutzer waren **Studenten**. Fast alle gaben eine **häufige Nutzung von GoogleMaps** und **viel Erfahrung mit Computern** an. Das Durchschnittsalter war **26 Jahre**, wobei hier von 20-32 Jahren alle Altersgruppen recht gleichmäßig vertreten waren (Abbildung 5.2).

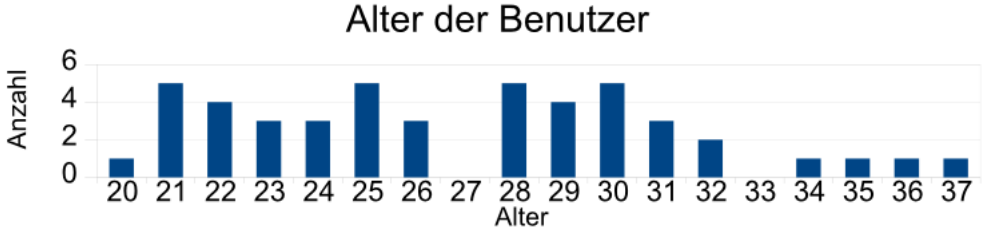

Abbildung 5.2: Alter der Benuzter

5.4 Heatmap

Als erste Darstellung wurde die Heatmap 3.3.3 angezeigt. Hier war es die Aufgabe des Benutzers, einen Wert zu einer bestimmten Uhrzeit möglichst genau abzulesen. Hierzu wurde die Legende mit dem Farbverlauf verwendet (Abbildung 3.2).

Der abzulesende Wert war 7°C. Die Benutzer schätzten ihn von 0°C bis 20°C, wobei knapp zwei Drittel genauer als 3°C am richtigen Wert war.

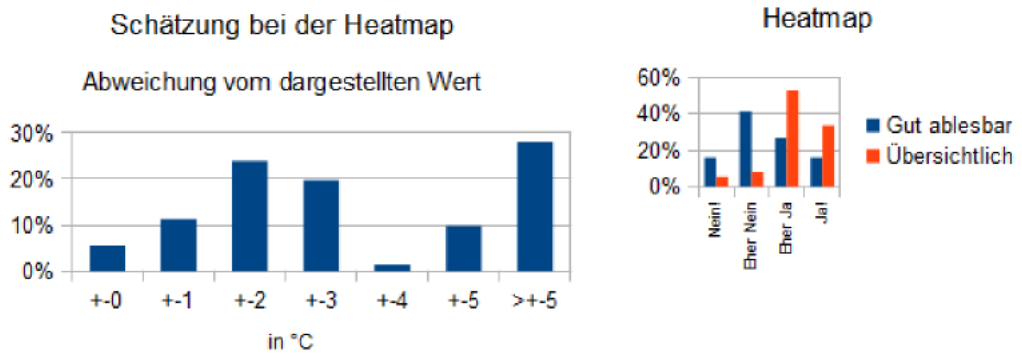

Abbildung 5.3: Evaluationsergebnisse der Heatmap

Hierbei ist zu beachten, dass bei der Darstellung der Legende ein Bug in einigen Chrome Browsern aufgetreten ist, bei dem diese nicht richtig angezeigt wurde. So könnte sich der hohe Anteil an Leuten erklären, die um mehr als 5°C daneben lagen. Der Durchnittswert aller Schätzungen war 7,78°C mit einer Standartabweichung von 5,1°C (Abbildung 5.3).

Kommentare
Einige Benutzer beklagten sich über **die Transparenz** der Heatmap, die das Ablesen erschwert. Gerade da der Untergrund durch beispielsweise Waldgebiete gleich gefärbt sein kann, kann es hier zu verfälschten Wahrnehmungen kommen. Auch wurde der Vorschlag gemacht, eine **stärkere Abstufung der Farben** einzubauen, so dass man die Temperaturbereiche besser erkennen kann. Dies kann auch durch die **Isolinien** erreicht werden, die zwar implementiert, aber nicht in der Evaluation verwendet wurden. Auch wurde bemängelt, dass die Legende zu klein sei. **Mouseover** einzubauen, um die genauen Werte an den jeweiligen Stellen der Heatmap angezeigt zu bekommen, war eine weitere Idee eines Benutzers.

Fazit
Die Werte der Heatmap konnten von vielen Benutzers einigermaßen genau abgelesen werden. Die Darstellung wurde als **sehr übersichtlich** bewertet, bei der man jedoch die Werte **eher nicht so gut ablesen** kann.

5.5 Animierte Heatmap

Danach wurde die animierte Heatmap vorgestellt, wobei die Benutzer die niedrigste und die höchste Temperatur zu einer anderen Uhrzeit schätzen sollten.

Die abzulesenden Werte waren -4°C und 2°C. Die Benutzer schätzten den niedrigen Wert von -12°C bis 10°C, wobei aber nur ein Fünftel genauer als 3°C am richtigen Wert war. Das Maximum schätzen die Benutzer von -5°C bis 26°C ein, wobei ein Drittel genauer als 3°C am abzulesenden Wert waren.

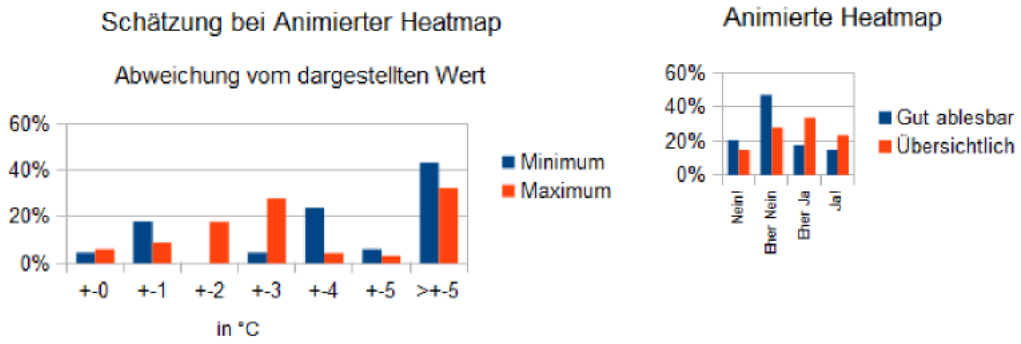

Abbildung 5.4: Evaluationsergebnisse der animierten Heatmap

Fazit:
Durch die Animation wird das Ablesen der Werte viel schwieriger. Auch stieg der Anteil der Leute, die die Darstellung als unübersichtlich deklarierten, im Gegensatz zur normalen Heatmap, deutlich an (Abbildung 5.4). Auch die Ablesbarkeit wurde eher schlechter bewertet als bei der Heatmap ohne Animation.

5.6 Liniendiagramm

Als nächste Darstellung sollte das Liniendiagramm bewertet werden. Dabei wurden Feinstaubkonzentration und Ozonbelastung dargestellt (Abbildung 5.5). Der Benutzer sollte hierbei die Ozonbelastung für alle Drittel der Route bewerten.

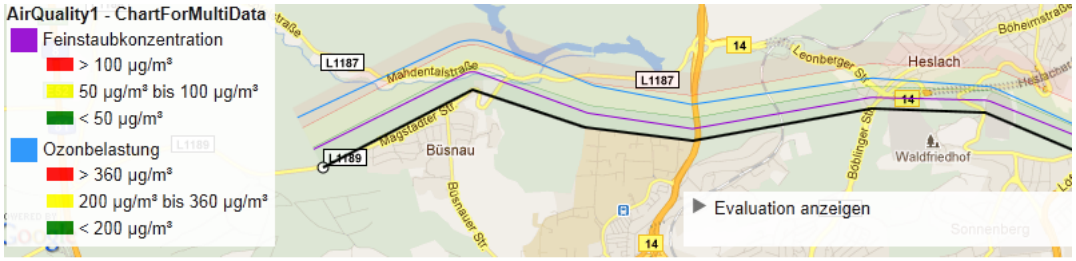

Abbildung 5.5: Verwendetes Liniendiagramm bei der Evaluation.

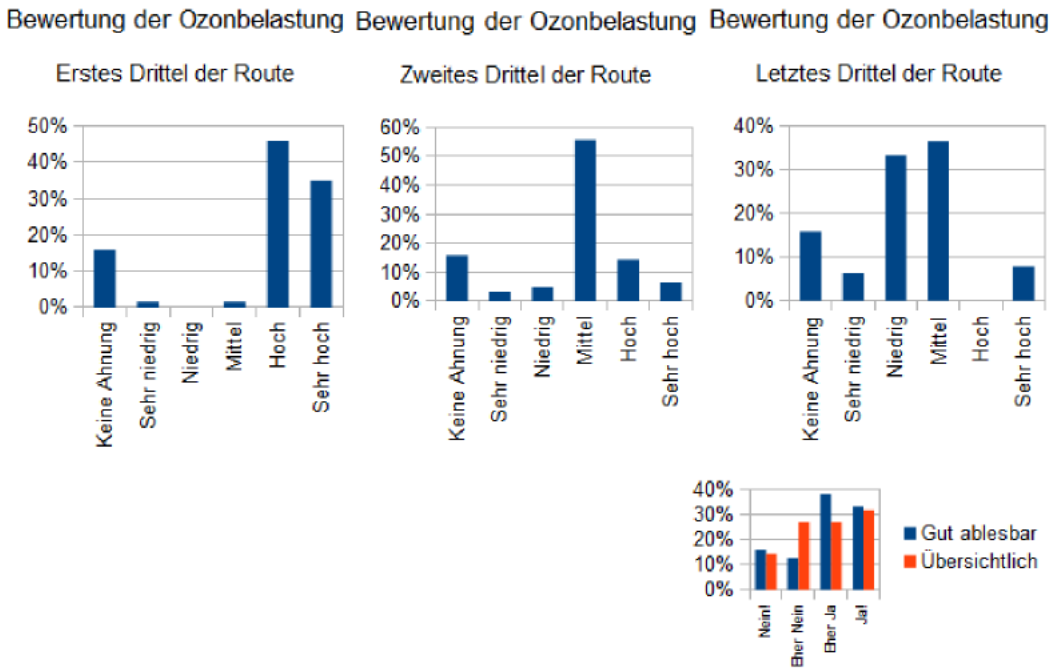

Abbildung 5.6: Evaluationsergebnisse des Liniendiagramms

Es fällt sofort auf, dass es hier sehr viele ähnliche Bewertungen der einzelnen Drittel der Routen waren. Dies zeigt, dass sehr viele Benutzer die Darstellung interpretieren konnten. Nur für **15% der Benutzer** schienen diese Darstellung nicht geeignet zu sein (Abbildung 5.6).

Bei der Bewertung der Übersichtlichkeit und Ablesbarkeit der Darstellung gab es geteilte Meinungen. Die Daten bewerteten jeweils ein Drittel der Benutzer als schlecht, gut bzw. sehr gut ablesbar und übersichtlich.

Kommentare
Viele Benutzer fanden den **Hintergrund zu hell** und schlugen vor, die **Grenzlinien** zwischen den Bereichen besser hervorzuheben. Auch **Achsenbeschriftungen** hätten manchen Benutzern hierbei gefallen. Ein Kritikpunkt war auch, dass sich **vertikale Verläufe** der Linien eventuell schlechter ablesen lassen könnten, als die horizontale Linien, die in der Evaluation hauptsächlich gezeigt wurden.

Fazit
Vielen Benutzern gefiel diese Darstellung. Vor allem sind die Werte für diesen Teil der Benutzer gut ablesbar. Eventuell müssen noch Feineinstellungen bei der Farbgebung und Transparenz vorgenommen werden, um die Bereiche besser sichtbar zu machen.

5.7 Wind

Es wurden insgesamt vier Winddarstellungen verglichen: Statischer Windpfeil 3.9 auf der Fläche und auf der Route, ein animierter Windpfeil 3.10 auf der Route und die Windpartikel 3.4. Dazu wurde jeweils wie beim Liniendiagrammtest gefragt, wie der Benutzer den Wind in den einzelnen Dritteln der Route bewertet. Zudem wurde noch die Frage gestellt, ob der Benutzer, anhand der hier dargestellten Winddaten, sich dafür entscheiden würde, die Strecke mit dem Fahrrad zu fahren.

Die Darstellungen wurden in zufälliger Reihenfolge gezeigt, damit eventuelle Lerneffekte während eines anderen Windtests den Vergleich der Winddarstellungen untereinander nicht so stark beeinflussen. Zudem wurden immer die gleiche Route mit den gleichen Winddaten dargestellt, sodass man die subjektiven Eindrücke der Darstellungen evaluieren kann.

5.7.1 Statischer Windpfeil auf der Fläche

Abbildung 5.7: Windpfeile auf der Fläche

Bei dieser Darstellung wurden statische Windpfeile mittels eines Flächenrenderer dargestellt (Abbildung 5.7).

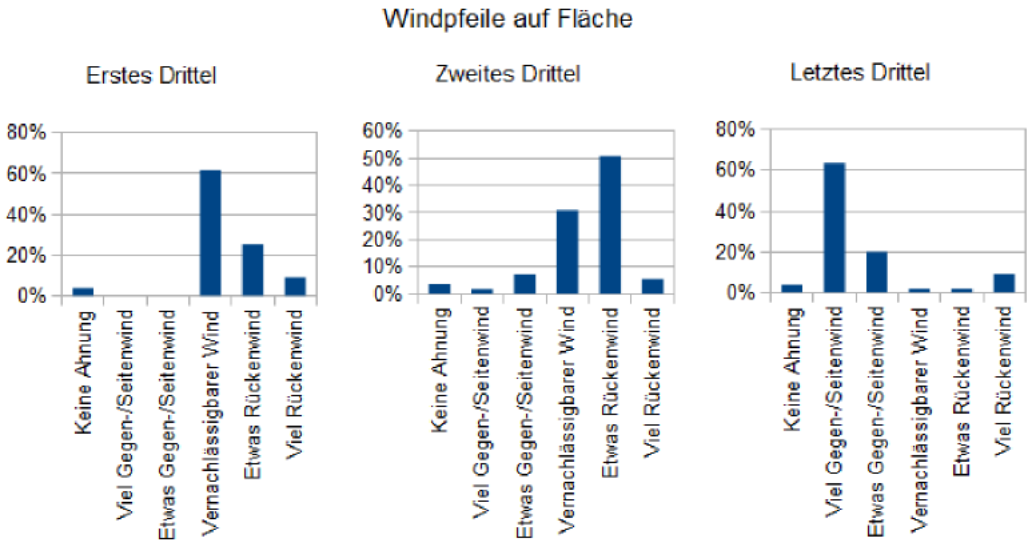

Abbildung 5.8: Evaluationsergebnisse des statischen Windpfeiles auf der Fläche

Ungefähr 50%-60% der Benutzer waren sich bei der Bewertung des Windes einig. Insgesamt fanden an die 70% diese Darstellung gut ablesbar und ebenfalls 70% übersichtlich (Abbildung 5.8).

Kommentare
Von einigen Benutzern wurde **eine Legende** für die Windstärke gefordert, um diese besser einschätzen zu können. Auch waren manchen Benutzern **die Pfeile zu breit**.

Fazit
Schmälere Windpfeile könnten kleinere Bereiche der Karte überdecken und so diese Darstellung noch weiter verbessern. Auch wenn sich in der Evaluation zeigte, dass die Benutzer ungefähr die gleiche Einschätzung ohne eine Legende und damit exakt ablesbare Windgeschwindigkeiten, hatten, ist eine Legende zu empfehlen. Es wären noch genauere Studien interessant, die untersuchen, ob Legenden für diesen Anwendungsfall überhaupt notwendig sind.

5.7.2 Statischer Windpfeil auf der Route

Abbildung 5.9: Windpfeile auf Route

Bei dieser Darstellung wurde der statische Windpfeil mittels eines Routenrenderer entlang der Route platziert (Abbildung 5.9).

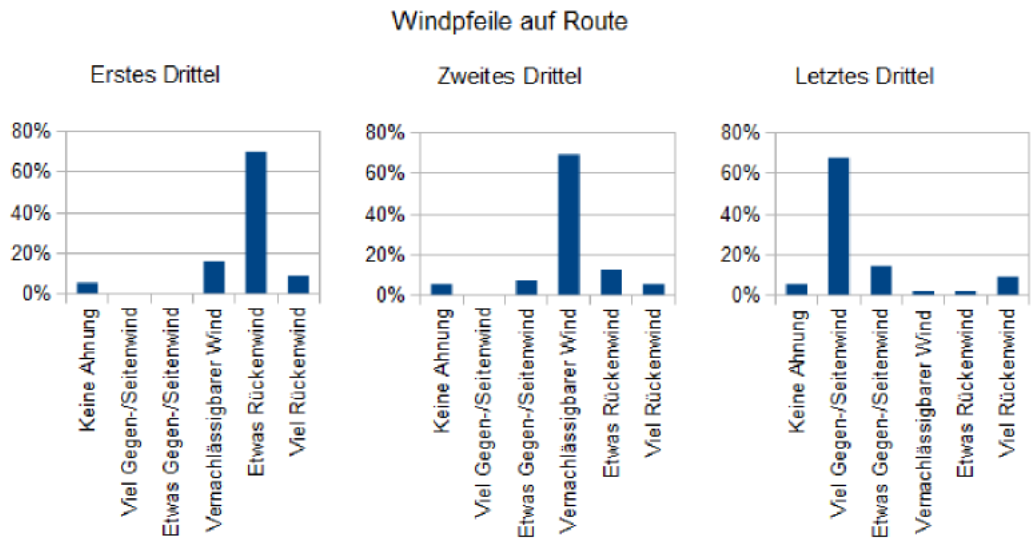

Abbildung 5.10: Evaluationsergebnisse des statischen Windpfeiles auf der Route

Hier gab es sehr einheitlichere Bewertungen des Windes. Auch wurde diese Darstellung am Übersichtlichsten bewertet, bei der die Daten am Besten eingeschätzt werden können (Abbildung 5.10).

Kommentare
Die **Pfeile wurden als schlecht erkennbar** deklariert, eventuell da diese im zweiten Drittel der Route zu klein sind. Dies ist aber auch beabsichtigt, da der Wind hier vernachlässigbar ist. Es wurde der Vorschlag gemacht, dass man die Pfeile **nur in der Länge variieren** könnte, so dass diese immer gut sichtbar sind. Auch hier fehlte einigen Benutzern eine **Legende** für die Windstärke.

Fazit
Die Idee, die Pfeile nur in der Länge zu verändern, klingt interessant und sollte noch weiter verfolgt werden. Aufgrund der sehr hohen Übersichtlichkeits- und Ablesbarkeitswerte scheint dies eine der besten Darstellung für den Wind zu sein.

5.7.3 Animierter Windpfeil auf der Route

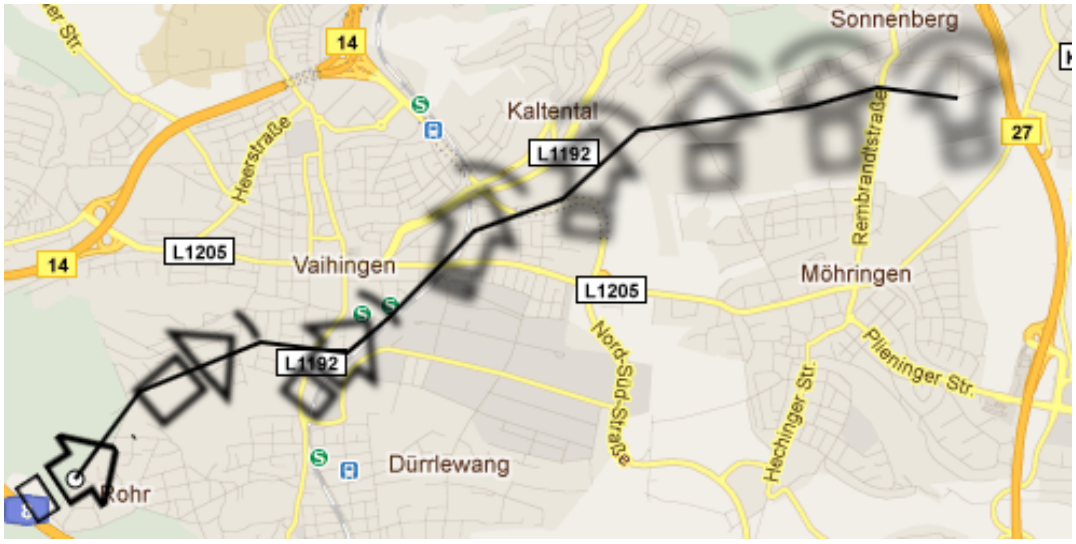

Abbildung 5.11: Animierte Windpfeile auf Route

Bei dieser Darstellung wurde ein animierter Windpfeil entlang der Route verwendet (Abbildung 5.11).

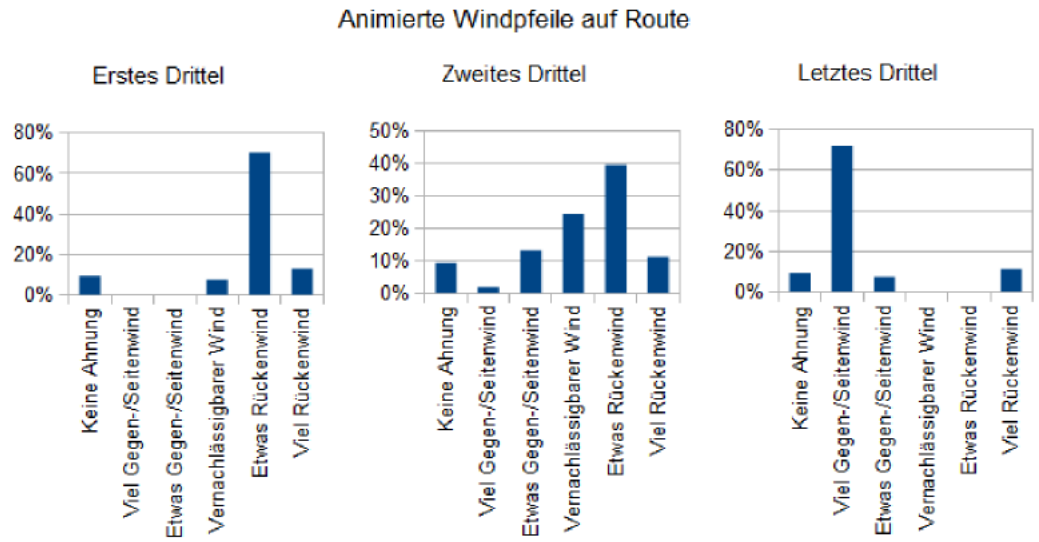

Abbildung 5.12: Evaluationsergebnisse des animierten Windpfeiles auf der Route

Vor allem im zweiten Drittel ist hier eine große Schwankung festzustellen. Dies könnte an der Animation liegen, die eine gewissen Mindestgeschwindigkeit hatte. Bei den anderen

Dritteln fiel die Bewertung dagegen mit ca. 80% gleichen Antworten sehr gut aus (Abbildung 5.12).

Die Darstellung wurde von vielen als übersichtlich und gut ablesbar bewertet.

Kommentare
Auch hier wurde die fehlende Legende bemängelt. Zudem wurde oft gesagt, dass viel Zeit nötig sei, um die Geschwindigkeit zu bewerten. Dies haben andere Benutzer damit begründet, dass man jeden Pfeil einzeln anschauen muss, um einen Eindruck über die Windstärke auf der Route bekommen zu können.

Fazit
Auch hier muss, wie bei den vorherigen Darstellungen, eine Legende eingefügt werden, was sich durch die Animation aber schwieriger umsetzen lässt als bei den statischen Pfeilen. Eventuell muss die Mindestgeschwindigkeit der Pfeile herabgesetzt werden, um vernachlässigbare Windgeschwindigkeiten deutlicher darzustellen. Auch kann der Ansatz von anderen Linien, wie bereits in Kapitel 3.3.11 beschrieben, verwendet werden, um dieses Problem zu mindern.

5.7.4 Windpartikel

Abbildung 5.13: Windpartikel

Bei dieser Darstellung wurden die Windpartikel verwendet (Abbildung 5.13).

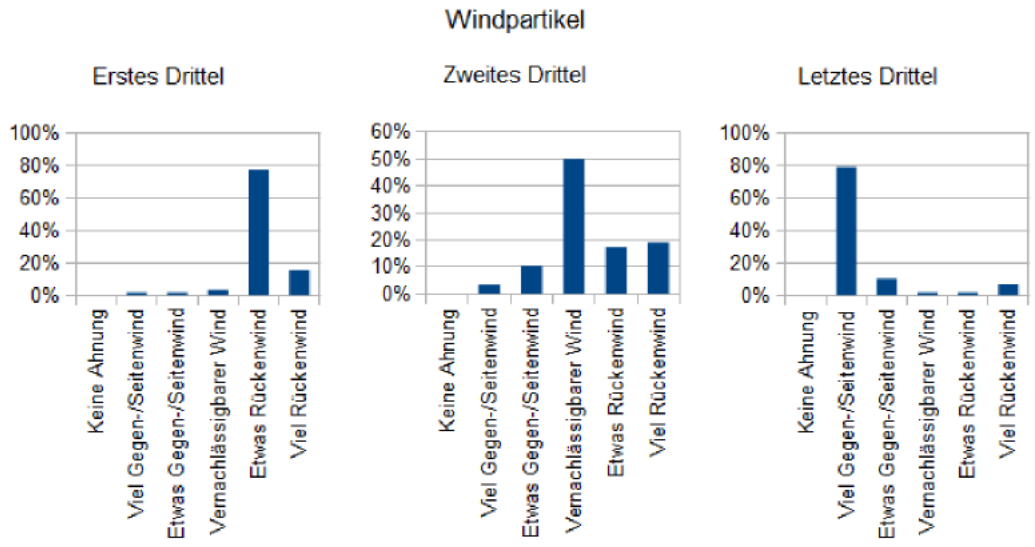

Abbildung 5.14: Evaluationsergebnisse der Windpartikel

Hier ist zu beachten, dass viele im zweiten Drittel „Viel Rückenwind“ angegeben haben. Im zweiten Drittel haben sich die Punkte aber kaum bewegt. Dies lässt vermuten, dass diese Benutzer den Zeitschieberegler nicht verwendet haben um bei dieser Darstellung in den anderen Bereich zu sehen. In den anderen Dritteln sind die Aussagen mit jeweils an die 80% sehr eindeutig ausgefallen, was dafür spricht, dass diese Darstellung gut zu interpretieren ist (Abbildung 5.14).

Kommentare
Eine Erklärung für dieses Verhalten könnte, wie ein Benutzer geschrieben hat, sein, dass die **Animation sehr verwirrend** sein kann, da obwohl sich die Teilchen bewegen nur ein Zeitpunkt abgebildet wird. Zudem schrieben zwei Benutzer, dass sie diese Darstellung als **sehr unruhig** empfanden. Auch fehlt hier wieder die **Skala** für die Windstärke.

Fazit
Der Schieberegler hätte bei dieser Darstellung noch ausdrücklich erwähnt werden sollen. Um diese Darstellung statischer zu machen, könnte man diese Linien an den Punkten eines Rasters „aufhängen“ und damit die Animation aus der Darstellung nehmen. Zudem könnte man dabei über ein dargestelltes Dreieck die Unsicherheit der Windrichtung noch mit anzeigen. Dies würde als eine neue Darstellung „Windfahne“ realisiert werden können. Auch lässt sich diese an der Route „befestigen“, so dass das Animieren über die Zeit wegfallen könnte.

5.7.5 Vergleich der Winddarstellungen

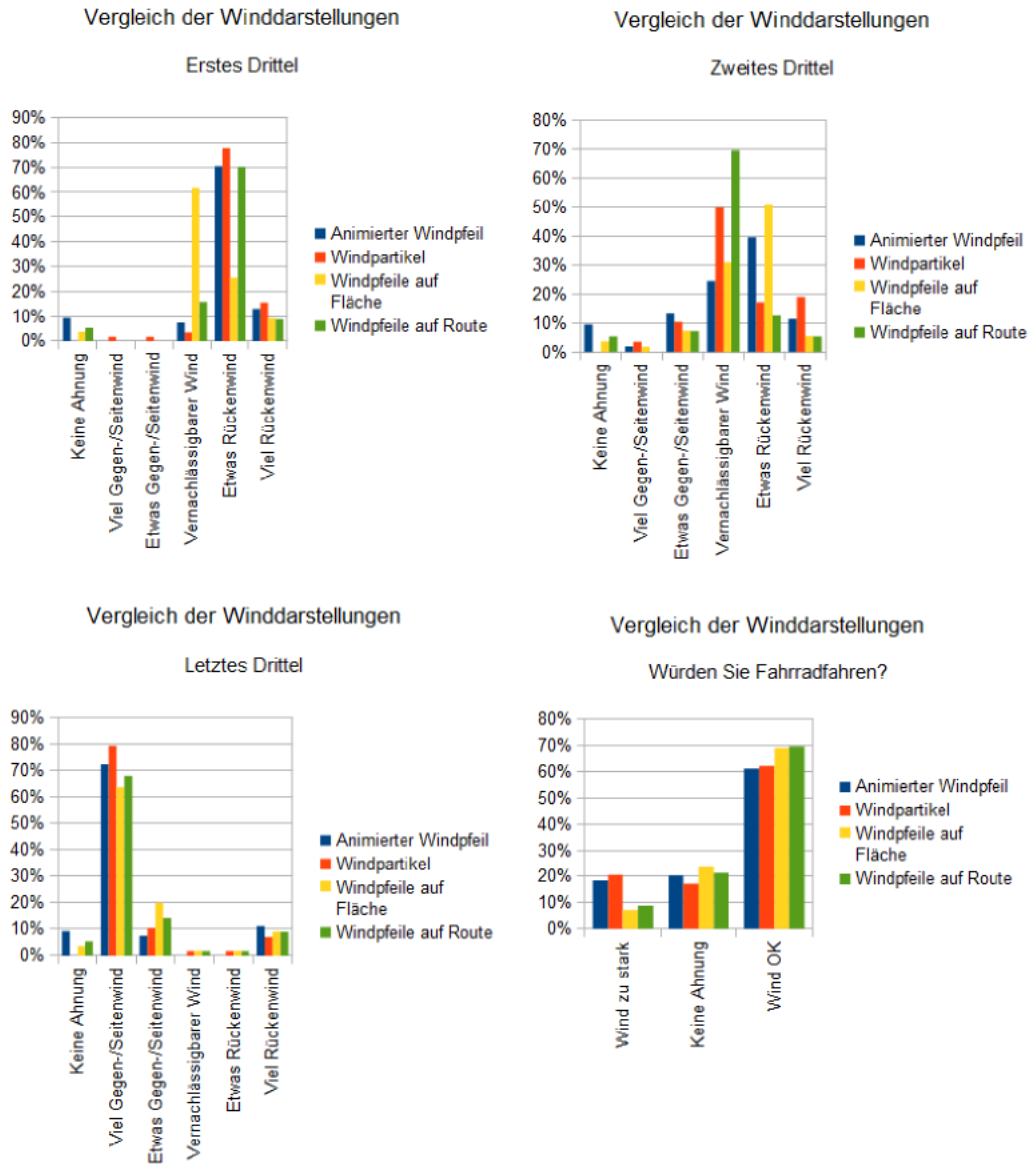

Abbildung 5.15: Vergleich aller Winddarstellungen

Im ersten Drittel fällt auf, dass der Wind bei Flächendarstellung anders erkannt wurde als bei allen anderen Darstellungen. Dies könnte daran liegen, dass man hier den Vergleich mit anderen Windpfeilen am Ende der Route hatte, die viel größer waren. Die Fahrradfahr-Frage bewerteten beim animierten Windpfeil und den Windpartikeln 10% mehr den Wind als zu stark im Vergleich zu den anderen Darstellungen (Abbildung 5.15).

5.8 Kombinierte Ansichten

Bei den kombinierten Ansichten wurden einige spezielle Fragen an den Benutzer gestellt, die er anhand der kombinierten Darstellung beantworten soll. Dies soll zeigen, welche Aspekte bei der Kombination für den Benutzer irritierend oder störend wirken. Zudem wurde noch ein Frageblock für alle kombinierten Ansichten angezeigt. Ziel der Darstellungen sollte es sein, Windrichtung, Windstärke, Luftqualitätsdaten und Temperatur gleichzeitig auf der Karte darzustellen. Daher wurde gefragt, welche dieser Faktoren die Benutzer gut einschätzen können. Am Ende wurde noch die Übersichtlichkeit der gesamten kombinierten Darstellung abgefragt.

Anders als bei den anderen Bereichen der Evaluation gab es hier anstatt nur eines Kommentarfeldes zwei Freitextfelder. Hier konnte der Benutzer eintragen, was ihn besonders gestört oder gefallen hat und wie man die Darstellung noch verbessern kann. Diese wurden von noch mehr Besuchern verwendet, als die Kommentarfelder bei der Einzeldarstellungen (Abbildung 5.16).

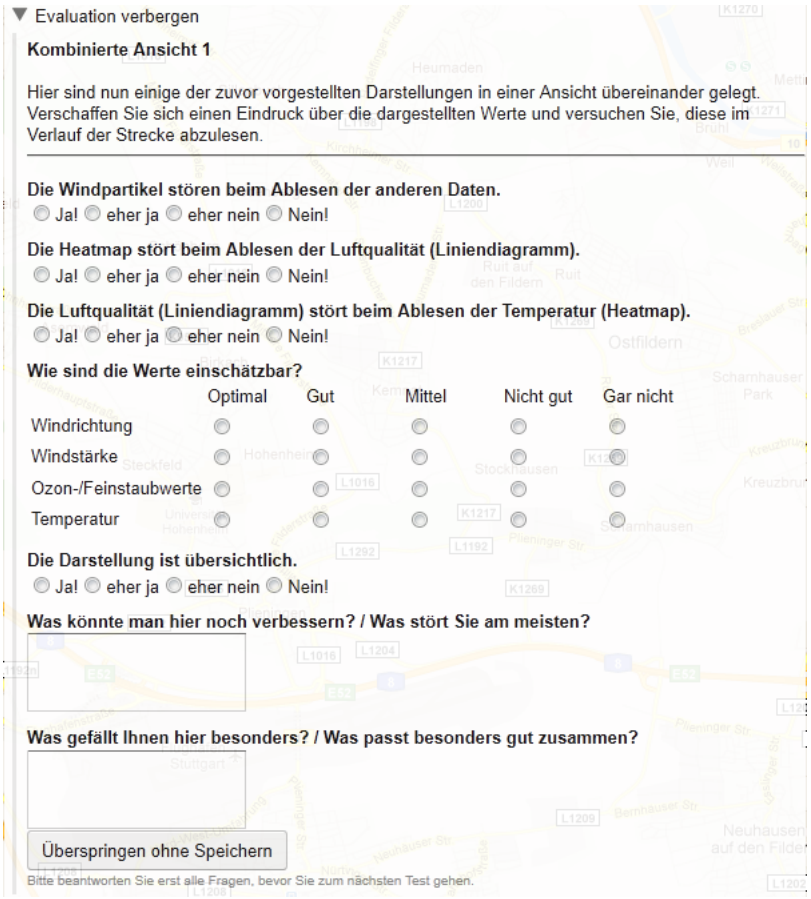

Abbildung 5.16: Evaluationsbeispiel einer kombinierten Ansicht

5.8.1 Erste kombinierte Ansicht

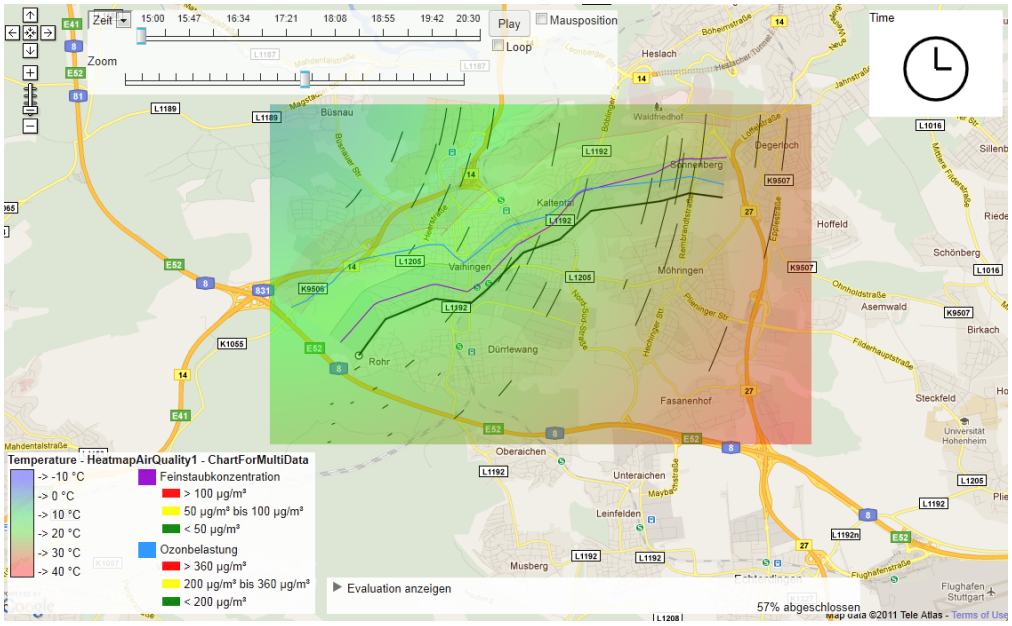

Abbildung 5.17: Die erste kombinierte Ansicht

Verwendete Darstellungen

- Windpartikel
- Heatmap für Temperatur
- Liniendiagramm für Luftqualität
- Uhricon auf separater Ansicht

Bei dieser kombinierten Ansicht sollte untersucht werden, ob die Heatmap in Kombination mit einem Liniendiagramm interpretiert werden kann. Dabei waren die kritischen Stellen die farblich gekennzeichneten Bereiche des Liniendiagramms zur Einteilung der drei Bereiche. Für den Wind wurden die Windpartikel verwendet und gefragt, ob diese störend wirken (Abbildung 5.17).

Zwei Drittel der Benutzer meinten, dass die Heatmap beim Ablesen des Liniendiagramms stört (Abbildung 5.18). Nur ein Drittel aber meinte, dass das Liniendiagramm beim Ablesen der Heatmap stört. Windpartikel stören dabei die Benutzer nicht. Die Winddaten waren, bis auf die fehlende Legende für die Windpartikel, gut einschätzbar (Abbildung 5.19).

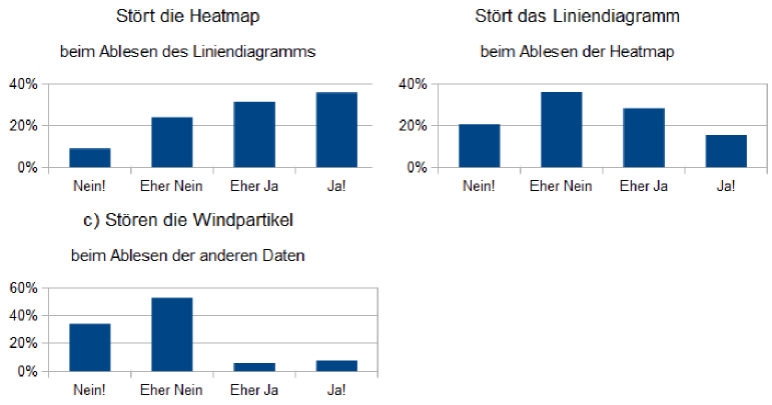

Abbildung 5.18: Die zusätzlichen Fragen zur ersten kombinierten Ansicht

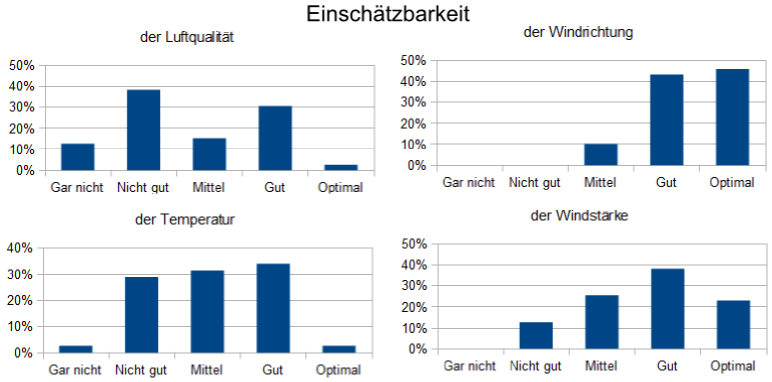

Abbildung 5.19: Die Ergebnisse zur Einschätzbarkeit der einzelnen Daten bei der ersten kombinierten Ansicht

Besonders gut war
In den Kommentaren wurden sehr oft die **Windpartikel gelobt**. Diese passen gut zur Heatmap und wären sehr gut erkennbar. Das Liniendiagramm war für viele Benutzer auch **nach kurzer Eingewöhnung gut verwendbar**.

Besonders schlecht war
Die überlappenden Farben waren der Hauptkritikpunkt der Benutzer. Auch fanden einige das Diagramm etwas zu überladen.

Fazit und Verbesserungsvorschläge
Durch Einfügen von Isolinien könnte die Heatmap noch besser abgelesen werden. Dadurch könnte sie aber noch mehr bei der Verwendung des Liniendiagramms stören. Falls man das Liniendiagramm noch besser ablesbar machen kann, könnte diese Darstellung noch Potenzial haben. Windpartikel scheinen für fast alle Benutzer eine gute Darstellung zu sein, die kaum Einfluss auf die Benutzbarkeit der anderen Darstellungen hat.

5.8.2 Zweite kombinierte Ansicht

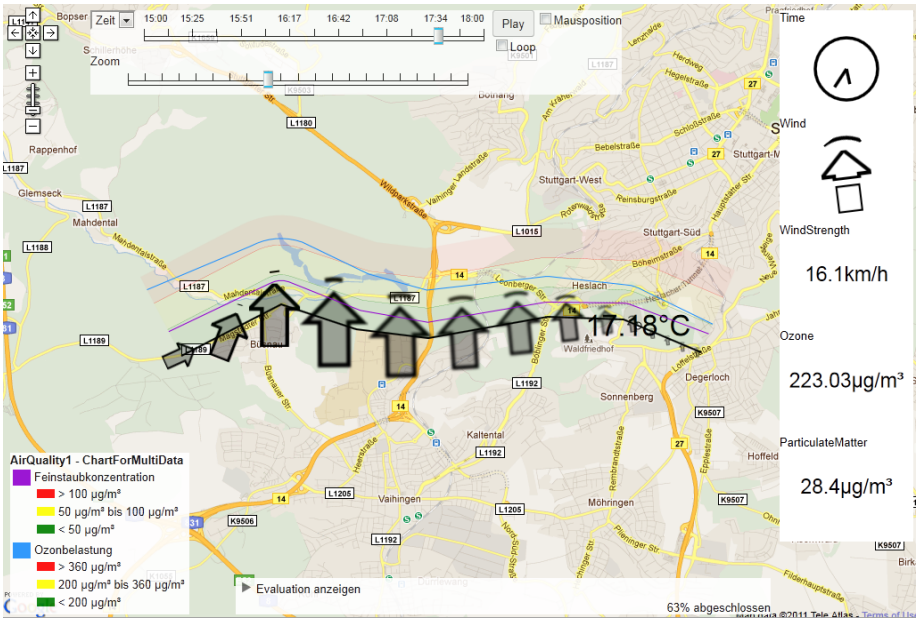

Abbildung 5.20: Die zweite kombinierte Ansicht

Verwendete Darstellungen

- Uhricon und animierter Windpfeil auf separater Ansicht
- Text mit Windstärke und Luftqualität auf separater Ansicht
- Liniendiagramm für Luftqualität
- Windpfeil auf Route
- Temperatur als Text am aktuellen Punkt

Um zu überprüfen, ob die Heatmap wirklich einen Einfluss auf die Ablesbarkeit des Liniendiagramms hat, wurde die Heatmap hier weggelassen und durch ein Texticon als Punktdarstellung ersetzt. Dies sollte auch zeigen, ob den Benutzern zum Ablesen der Temperatur dies lieber ist als die Heatmap. Zudem wurden die Luftqualitätswerte zusätzlich seitlich eingefügt. Der Wind wurde als Routendarstellung mit statischen Pfeilen angezeigt (Abbildung 5.20).

Recht eindeutig stimmen die Benutzer überein, dass die Textanzeige als separate Ansicht eine sinnvolle Ergänzung darstellt (Abbildung 5.21). Die Kombination der Texticon Punktdarstellung mit den Windpfeilen war aufgrund des fehlenden Kontrastes nicht gut ablesbar. Die Windpfeile störten aber die meisten Benutzer eher nicht beim Verwenden des Liniendiagramms.

Im Vergleich mit der ersten kombinierten Ansicht fällt auf, dass die Winddaten von noch mehr Benutzern als sehr gut einschätzbar angesehen werden (Abbildung 5.22). Die Luftqualität wird nun ebenfalls zum Großteil als gut ablesbar eingeschätzt. Selbst die Temperaturanzeige, die nur durch das Texticon am aktuellen Punkt dargestellt wurde, wurde mehrheitlich als gut ablesbar empfunden.

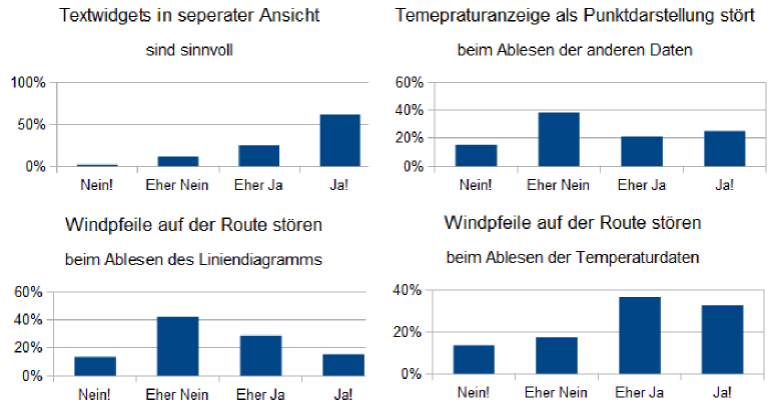

Abbildung 5.21: Die zusätzlichen Fragen zur zweiten kombinierten Ansicht

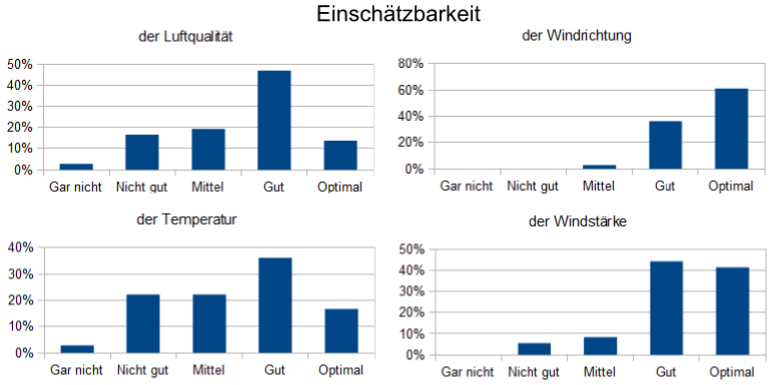

Abbildung 5.22: Die Ergebnisse zur Einschätzbarkeit der einzelnen Daten bei der zweiten kombinierten Ansicht

Besonders gut war
Viele Benutzer lobten die Anzeige der exakten Daten als separate Ansicht und die sehr übersichtlichen Windpfeile.

Besonders schlecht war
Missfallen hat einigen Benutzern, dass man die genauen Daten rechts braucht und Benutzer, die keine Experten sind, keinen Vergleich haben, obwohl dafür das Liniendiagramm gut geeignet schien. Viele wollten eine Art Ampeldarstellung, wie sie in der nächsten kombinierten Ansicht evaluiert wird.

Fazit und Verbesserungsvorschläge
Die Temperatur als Text sollte sich besser abgrenzen, um sie besser lesbar zu machen. Auch wurde vorgeschlagen die Temperatur nur in der separaten Ansicht darzustellen. Es fällt hier wieder auf, dass Routendarstellungen besonders gut bei den Benutzern ankommen.

5.8.3 Dritte kombinierte Ansicht

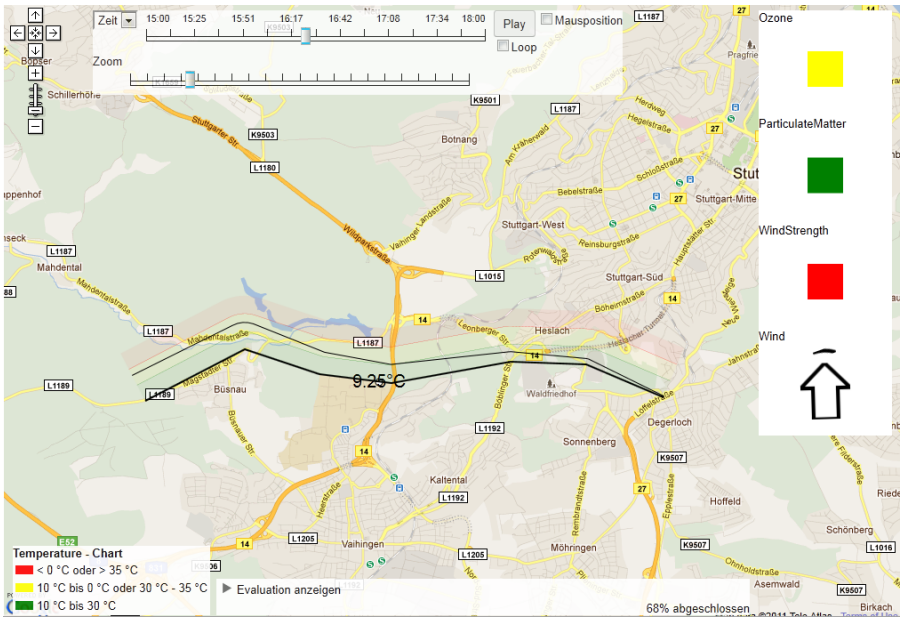

Abbildung 5.23: Die dritte kombinierte Ansicht

Verwendete Darstellungen

- Animierter Windpfeil auf separater Ansicht
- Ampel mit Windstärke und Luftqualität auf separater Ansicht
- Liniendiagramm für Temperatur
- Temperatur als Punkt

Die zuvor verwendeten Textanzeigen in separater Ansicht wurden nun durch Ampelicons ersetzt. Dies sollte zeigen, ob den Benutzern die Ampelwerte ausreichen oder ob sie die genauen Werte wollen. Zudem wurde die Temperatur als Liniendiagramm angezeigt (Abbildung 5.23).

Viele Benutzer fanden die Ampel als sinnvoll, aber nicht ausreichend (Abbildung 5.24). Die Temperatur als Liniendiagramm wurde von zwei Dritteln der Benutzer eher abgelehnt. Die Windrichtung, die nur durch einen Windpfeil in der separaten Ansicht dargestellt wurde, fanden über die Hälfte der Benutzer dennoch als gut einschätzbar (Abbildung 5.25). Die

Windstärke dagegen, welche im animierten Windpfeil und als Ampel dargestellt war, war eher mittelmäßig.

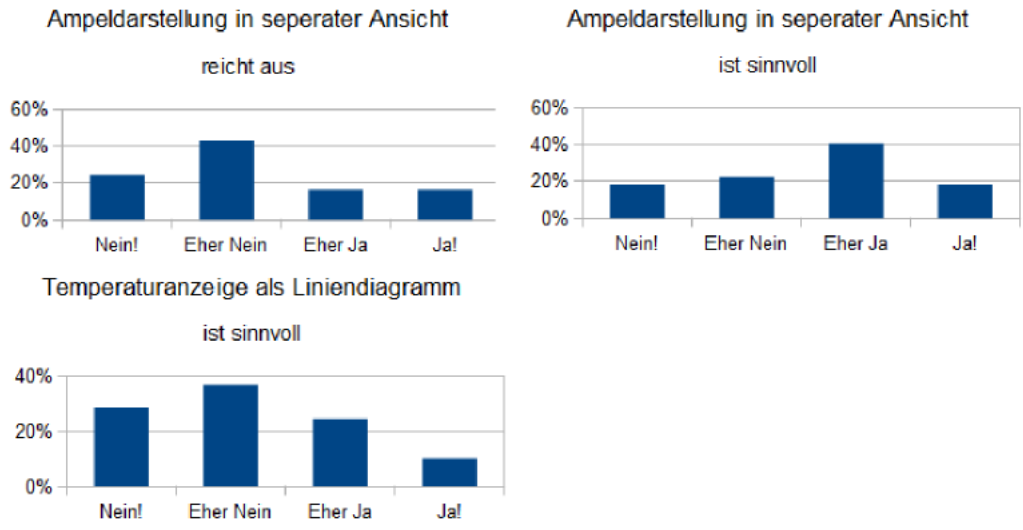

Abbildung 5.24: Die zusätzlichen Fragen zur dritten kombinierten Ansicht

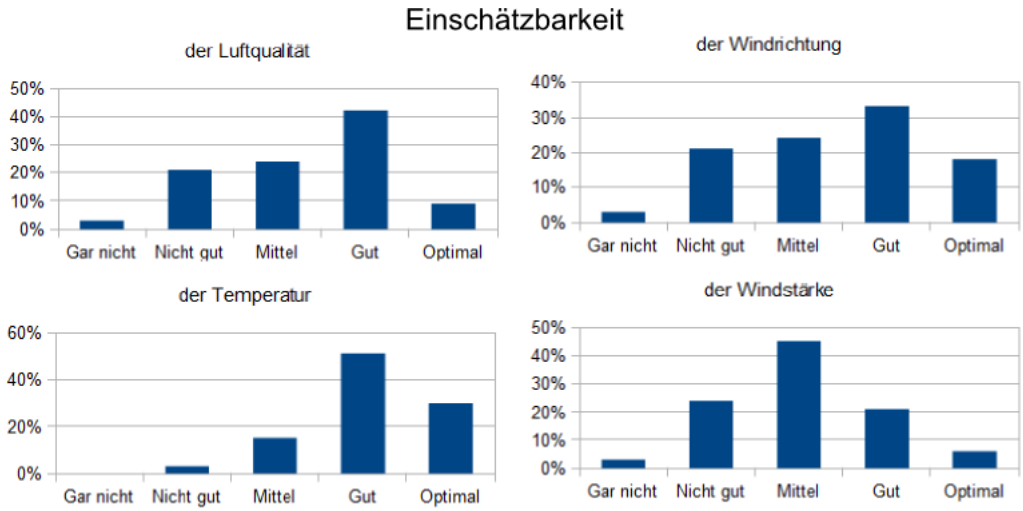

Abbildung 5.25: Die Ergebnisse zur Einschätzbarkeit der einzelnen Daten bei der dritten kombinierten Ansicht

Besonders gut war
Sehr viele Benutzer betonten hier nochmals, dass ihnen die **Ampelwerte ausreichen**.

Besonders schlecht war
Sehr viele Benutzer bemängelten, dass die **Ampelwerte nicht ausreichen** würden. Auch der Temperaturtext war für manche Benutzer an einigen Stellen schlecht ablesbar. Viele Benutzer beklagten sich auch über die **ungewohnte Temperaturskala** bei dem Liniendiagramm.

Fazit und Verbesserungsvorschläge
Die Ampelicons reichen allein nicht aus. Zusätzlich sollten, wie in der zweiten kombinierten Ansicht, die exakten Werte dargestellt werden. Die Windpfeile scheinen auch auf der Karte deutlich besser zu sein. Der Temperaturtext gehört nach Meinung einiger Benutzer auch in die separate Ansicht. Aufgrund der ungewohnten Temperaturskala, die lediglich die Temperatur bewertet statt anzeigt, scheint auch nicht gut geeignet.

5.8.4 Vierte kombinierte Ansicht

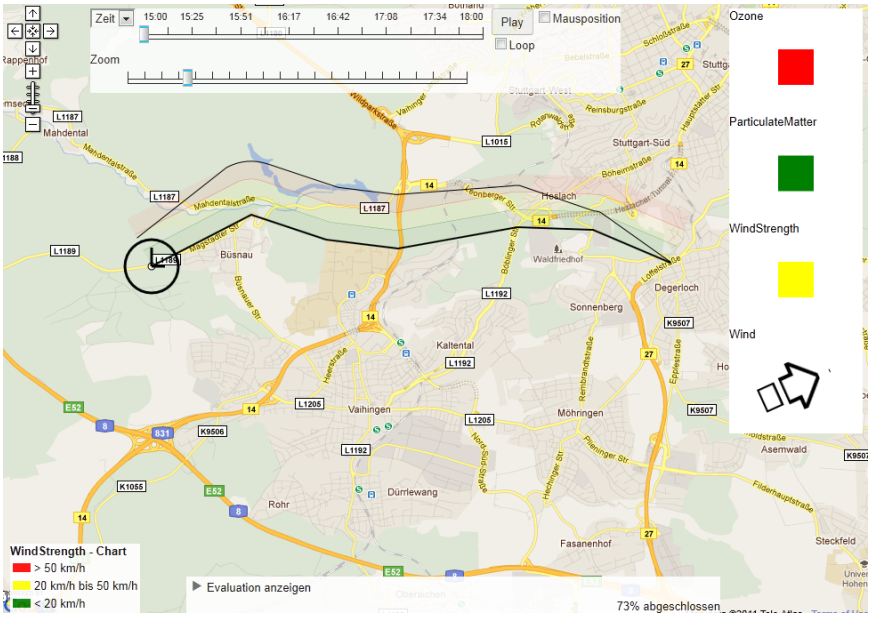

Abbildung 5.26: Die vierte kombinierte Ansicht

Verwendete Darstellungen

- Animierter Windpfeil auf separater Ansicht
- Ampel mit Windstärke und Luftqualität auf separater Ansicht
- Liniendiagramm für Windstärke
- Uhricon als Punkt

Als letzte Darstellung wurde die Uhr als Punkticon dargestellt und die Windstärke auf dem Liniendiagramm dargestellt. Dabei wird die für diesen Werte entscheidende Windrichtung nur als Windpfeil in der separaten Ansicht dargestellt (Abbildung 5.26).

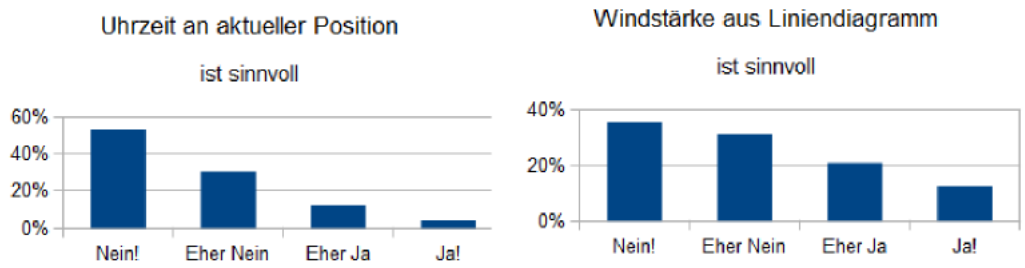

Abbildung 5.27: Die zusätzlichen Fragen zur vierten kombinierten Ansicht

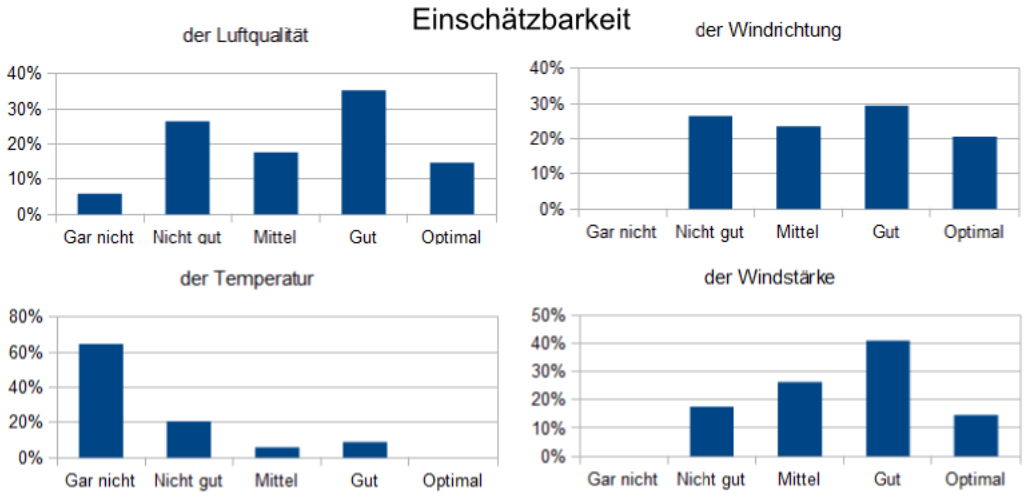

Abbildung 5.28: Die Ergebnisse zur Einschätzbarkeit der einzelnen Daten bei der vierten kombinierten Ansicht

Die Uhr an der aktuellen Position wurde fast komplett von den Benutzern abgelehnt (Abbildung 5.27). Das Liniendiagramm scheint auch nicht zur Anzeige der Windstärke geeignet (Abbildung 5.28).

Bei dieser Darstellung gab es recht wenig Kommentare, die hauptsächlich meinten, dass **die Uhr besonders schlecht als Punktdarstellung** sei. Auch dass die **Temperatur gar nicht dargestellt** war, wurde bemängelt.

Fazit und Verbesserungsvorschläge
Als Fazit aus dieser Bewertung kann festgestellt werden, dass die Uhr in die separate Darstellung muss.

5.8.5 Kombinierte Ansichten im Vergleich

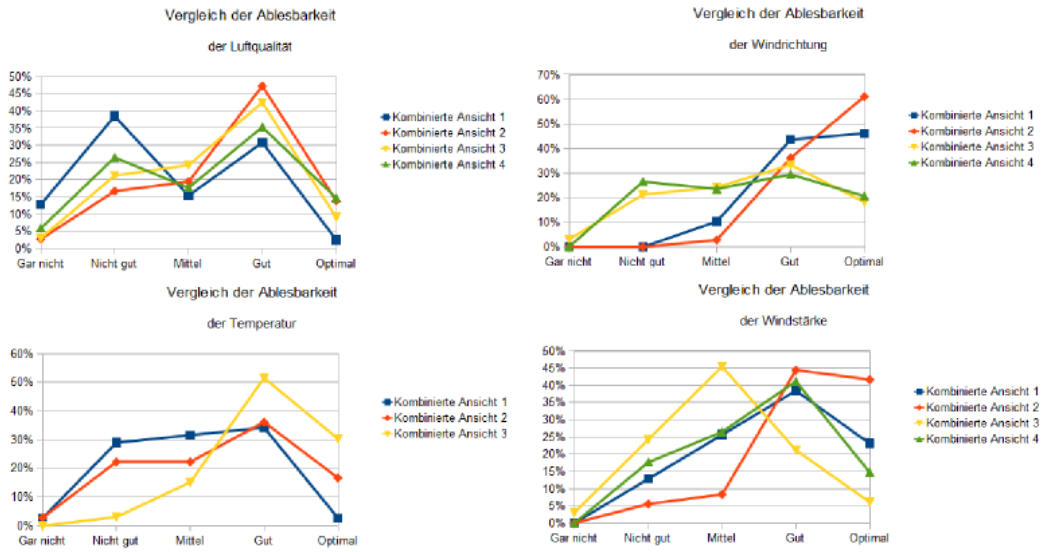

Abbildung 5.29: Ablesbarkeiten der einzelnen Daten in den kombinierten Ansichten im Vergleich

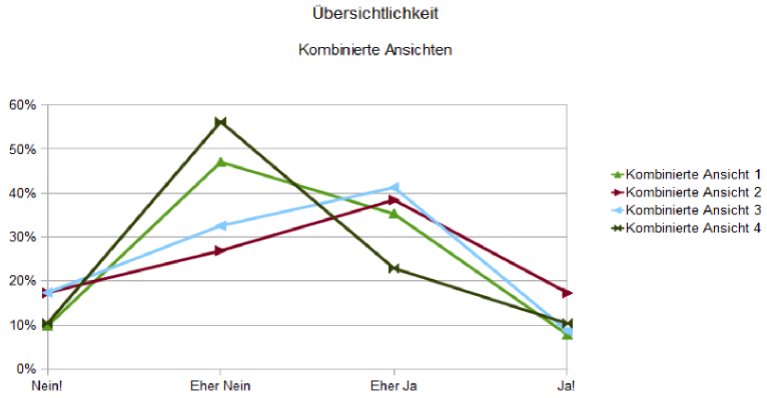

Abbildung 5.30: Allgemeine Übersichtlichkeit der kombinierten Ansichten im Vergleich

Die Temperatur wurde in der dritten Darstellung, bei dem diese mit einem Texticon am aktuellen Ort und als Liniendiagramm dargestellt wurde, am besten bewertet (Abbildung 5.29). Die ablesbarkeit der Luftqualität wurde in der zweiten kombinierten Ansicht, welche dort als Liniendiagramm und in der separaten Ansicht dargestellt wurde, am besten bewertet. Die dort verwendeten statischen Windpfeile auf der Route wurden ebenfalls als am besten ablesbar empfunden. Die zweite Darstellung wurde zudem als die übersichtlichste bewertet (Abbildung 5.30).

5.9 Alle Ansichten im Vergleich

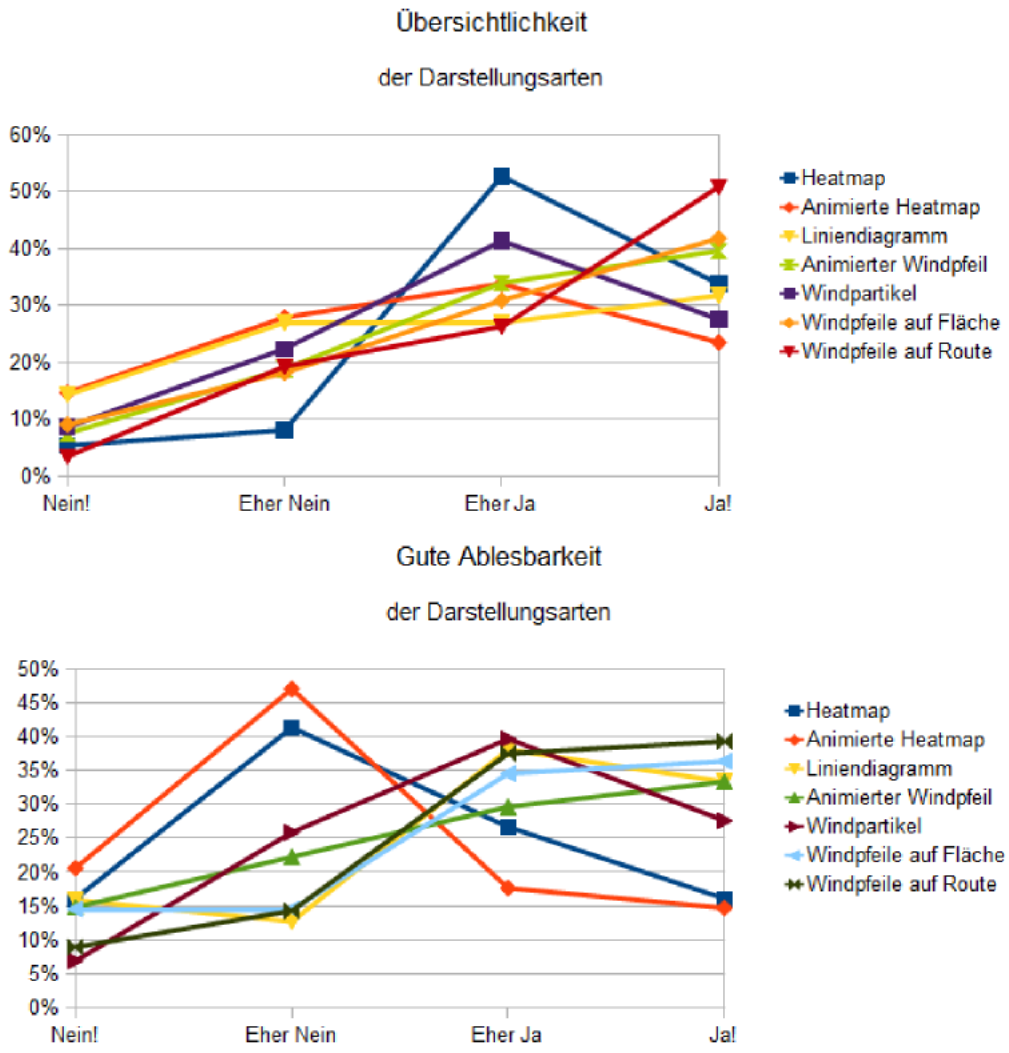

Abbildung 5.31: Übersichtlichkeit aller Darstellungen im Vergleich

Besonders übersichtlich und ablesbar sind mit Abstand die Windpfeile auf der Route. Die Heatmaps scheinen zwar übersichtlich zu sein, lassen sich aber im Vergleich mit den anderen Darstellungen schwer ablesen. Das Liniendiagramm wird zwar als eher nicht so übersichtlich beschrieben, dafür belegt es bei der Ablesbarkeit der Daten einen der vorderen Plätze (Abbildung 5.31).

5.10 Zusätzliche Fragen

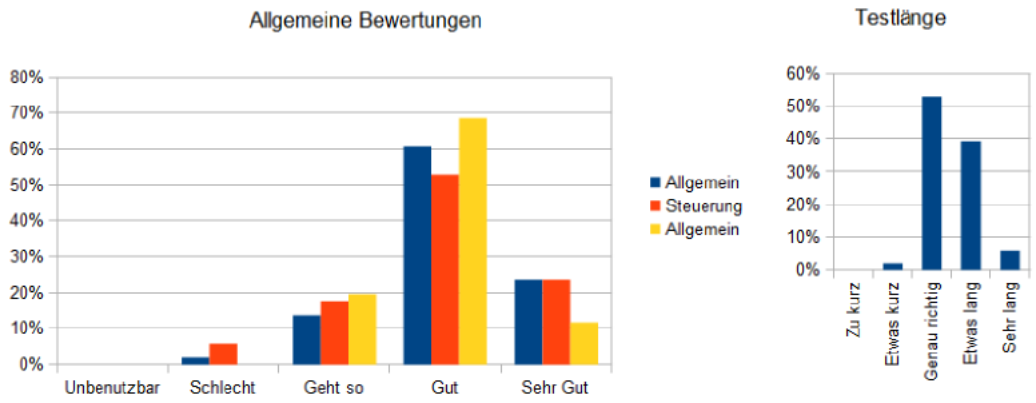

Abbildung 5.32: Bewertungen des Frameworks und Tests

Das Framework und die Steuerung wurden überwiegend positiv bewertet (Abbildung 5.32). Die Testlänge war nach Aussage der Benutzer gerade noch in Ordnung. Die Evaluation selbst gefiel den Benutzern ebenfalls.

Allgemein gefallen
Projekt und Framework
Die Idee des Projekts wurde häufig gelobt und für interessant befunden. Auch die Bedienelemente des Frameworks wurden gelobt.

Der Test
Am Test gefiel vor allem der direkte Vergleich der Anzeigemöglichkeiten.

Darstellungen
Vor allem die Routendarstellungen gefielen einigen Benutzern, da man hierbei nicht extra über die Zeit scrollen muss.

Allgemein missfallen
Projekt und Framework
Am Framework wurde von einigen Benutzern die Performance einiger Darstellungen bemängelt.

Darstellungen
Die analoge Uhr gefiel einigen Benutzern gar nicht. Auch die Flächendarstellungen wurden als nicht sinnvoll beschrieben.

5.11 Fazit der Evaluation

5.11.1 Separate Ansicht

Die Separate Ansicht wurde von vielen Benutzern allgemein als sinnvolle Ergänzung empfunden. Hier sollten Ampelicons mit den exakten Werten kombiniert immer zusätzlich zu anderen Darstellungen angezeigt werden, da die Icons in der Separaten Ansichten die anderen Darstellungen nicht stören und eine Alternative zu eventuell schlecht abzulesenden Werten darstellen. Von der alleinigen Anzeige in separaten Darstellungen sollte Abstand genommen werden, da die Benutzer immer zwischen dem Routenpunkt und dem separaten Widget hin und her schauen müssen, um die Werte zuzuordnen.

5.11.2 Liniendiagramm

Da die Werte aus dem Liniendiagramm für den Großteil der Benutzer gut abgelesen werden konnten, sollte diese Darstellung weiterverfolgt werden. Da die Übersichtlichkeit eher schlecht bewertet wurde, sollte hier angesetzt werden. Für Temperaturdaten scheint es eher nicht geeignet zu sein, da die Ampelskala hierbei mit dem erwarteten Werteverlauf nicht übereinstimmt. Winddaten lassen sich nur für das Liniendiagramm verwenden, wenn die Windrichtung mit in die Bewertung einfließt, sprich Gegen- und Rückenwind unterschiedlich betrachtet wird. Am Besten eignen sich demnach Luftqualitätsdaten für diese Darstellung.

5.11.3 Routendarstellungen

Routendarstellungen scheinen am besten geeignet zu sein, um die Werte entlang einer Route abzulesen. Dabei haben sie den Vorteil, dass man nicht durch die Zeit scrollen muss.

5.11.4 Neue Darstellungen

An neuen Darstellungen sind die Windpfeile zu erwähnen, die bei steigender Windstärke nur länger werden. Damit hat man immer gut sichtbare Pfeile, die nicht animiert werden müssen.

Auch die Windfahne ist eine neue Darstellung, die noch nicht betrachtet wurde. Dabei werden je nach Windstärke unterschiedlich lange Strecken an der Route befestigt, die in Windrichtung gedreht sind. Diese sehen ähnlich wie die Windpartikel aus und haben deren Vorteile. Zudem können die Windfahnen als Routendarstellung verwendet werden und bieten somit die zuvor genannten Vorteile.

6 Fazit

6.1 Zusammenfassung

Ziel dieser Arbeit war es, mögliche Darstellungen von Wetterdaten zu suchen, analysieren und deren Kombinationen zu untersuchen.

Nachdem einige Darstellungsmöglichkeiten gefunden wurden, wurde ein Framework implementiert, das mit Hilfe von SVG, GWT und GoogleMaps diese Daten darstellen kann. Dieses war dann die Grundlage für ein Tool, mit dem die Darstellungen und einige Kombinationen evaluiert werden konnten.

Die Evaluation lief sehr gut, mit vielen konstruktiven Kommentaren der Benutzer. Es gab sowohl einige überraschende Erkenntnisse wie auch erfüllte Prognosen. Auch Ideen für neue Darstellungen und neue Fragen für spätere Evaluationen sind ein Ergebnis dieser Evaluation.

6.2 Kombinationsmöglichkeiten

Unter anderem schien das Liniendiagramm und die Windpartikel in Kombination mit einer Heatmap gut geeignet zu sein, um Luftqualität, Wind und Temperatur gleichzeitig darzustellen. In der Evaluation stellte sich heraus, dass die Überlagerung der Farben aber zu viele Schwierigkeiten beim Ablesen bereitet.

Für Winddaten ist die Routendarstellung mit statischen Pfeilen zu empfehlen, da hier erstens nur die relevanten Windpfeile sichtbar und zweitens alles, ohne durch die Zeit scrollen zu müssen, auf einen Blick sichtbar ist. Die Pfeile wurden von den Benutzern als eher nicht störend in Bezug auf das Liniendiagramm, bewertet. Da die Windpartikel auch sehr gut ankamen, wären die hier nicht untersuchten Windfahnen noch eine Alternative, die viel Aussicht auf Erfolg verspricht.

Die zusätzliche Anzeige von Ampelicons und exakten Werten in der separaten Ansicht ist eine sehr gute Ergänzung und scheint, unabhängig von den anderen Darstellungen, immer verwendet werden zu können.

6.3 Ausblick

Für eine Folgearbeit wären die genannten Erweiterungen und eine damit verbundene neue Evaluation interessant.

Die Performance des Frameworks zu steigern ist ebenfalls nötig, um die Software praxistauglich zu machen. Dies passiert zum Teil durch die Browserhersteller selbst, die ihre JavaScript und Renderingperformance in den letzten Jahren drastisch steigern konnten. Ebenfalls bieten neue Technologien wie HTML5-Webworker in naher Zukunft noch die Möglichkeit, die Berechnungen zu parallelisieren.

Literaturverzeichnis

[BDW+08] Butkiewicz, Thomas ; Dou, Wenwen ; Wartell, Zachary ; Ribarsky, William ; Chang, Remco: Multi-Focused Geospatial Analysis Using Probes. In: *IEEE Trans. Vis. Comput. Graph.* 14 (2008), Nr. 6, S. 1165–1172 (Zitiert auf Seite 18)

[BW08] Byron, Lee ; Wattenberg, Martin: Stacked Graphs - Geometry & Aesthetics. In: *IEEE Trans. Vis. Comput. Graph.* 14 (2008), Nr. 6, S. 1245–1252 (Zitiert auf den Seiten 21 und 25)

[CR00] Cedilnik, Andrej ; Rheingans, Penny: Procedural annotation of uncertain information. In: *Proceedings of the conference on Visualization '00*. Los Alamitos, CA, USA : IEEE Computer Society Press, 2000. – ISBN 1–58113–309–X, S. 77–83 (Zitiert auf den Seiten 20 und 22)

[Dah06] Dahm, Markus: *Grundlagen der Mensch-Computer-Interaktion.* Pearson Studium, 2006. – 47–53 S. – ISBN 978–3–8273–7175–1 (Zitiert auf Seite 11)

[EKHW07] Eccles, Ryan ; Kapler, Thomas ; Harper, Robert ; Wright, William: Stories in GeoTime. In: *IEEE Symposium on Visual Analytics Science and Technology, 2007. VAST 2007.*, 2007, 19–26 (Zitiert auf Seite 21)

[Fis07] Fisher, Danyel: Hotmap: Looking at Geographic Attention. In: *IEEE Trans. Vis. Comput. Graph.* 13 (2007), Nr. 6, S. 1184–1191 (Zitiert auf den Seiten 21 und 22)

[FR81] Flury, Bernhard ; Riedwyl, Hans: Graphical Representation of Multivariate Data by Means of Asymmetrical Faces. In: *Journal of the American Statistical Association* 76 (1981), Nr. 376, S. 757–765. – ISSN 01621459 (Zitiert auf Seite 32)

[Geg06] Gegenfurtner, Karl R. ; Karnath, Hans-Otto (Hrsg.) ; Thier, Peter (Hrsg.): *Farbwahrnehmung und ihre Störungen*. Springer Berlin Heidelberg, 2006 (Springer-Lehrbuch). – 33–40 S. – ISBN 978–3–540–28449–9 (Zitiert auf Seite 11)

[GHJV95] Gamma, Erich ; Helm, Richard ; Johnson, Ralph ; Vlissides, John: *Design patterns: Elements of reusable object-oriented software*. Boston, MA, USA : Addison-Wesley Longman Publishing Co., Inc., 1995. – ISBN 0–201–63361–2 (Zitiert auf Seite 12)

[GS06] Grigthe, Henning ; Schumann, Heidrun: The Visualization of Uncertain Data: Methods and Problems. In: Schulze, Thomas (Hrsg.) ; Horton, Graham (Hrsg.) ; Preim, Bernhard (Hrsg.) ; Schlechtweg, Stefan (Hrsg.): *SimVis*, SCS Publishing House e.V., 2006. – ISBN 3–936150–46–X, S. 143–156 (Zitiert auf den Seiten 16, 17 und 20)

[HBO10] Heer, Jeffrey ; Bostock, Michael ; Ogievetsky, Vadim: A Tour through the Visualization Zoo. In: *ACM Queue* 8 (2010), 20:20–20:30. http://doi.acm.org/10.1145/1794514.1805128. – ISSN 1542–7730 (Zitiert auf Seite 20)

[Hom07] Hommen, Nils: Mashups und weborientierte Architekturen als Technologie-Trends des Web 2.0. In: Kollmann, Tobias (Hrsg.) ; Häsel, Matthias (Hrsg.): *Web 2.0: Trends und Technologien im Kontext der Net Economy* Bd. 1, Deutscher Universitätsverlag, 2007. – ISBN 978–3835008366, S. 103+ (Zitiert auf Seite 33)

[HT06] Hengl, Tomislav ; Toomanian, Norair: Maps are not what they seem: representing uncertainty in soil-property maps. In: Caetano, M. (Hrsg.) ; Painho, M. (Hrsg.): *7th International Symposium on Spatial Accuracy Assessment in Natural Resources and Environmental Sciences*, 2006, S. 805+ (Zitiert auf den Seiten 19 und 22)

[KKEM10] Keim, Daniel ; Kohlhammer, Jörn ; Ellis, Geoffrey ; Mansmann, Florian: *Mastering the Information Age - Solving problems with Visual Analytics*. Eurographics Association, 2010. – ISBN 978–3–905673–77–7 (Zitiert auf den Seiten 16 und 17)

[LP09] Luo, Xiaoxia ; Pan, Zhongying: Isoline Plotting Method of Discrete Geophysical and Geochemical Data. In: *Second InternationalSymposium on Knowledge Acquisition and Modeling, 2009. KAM '09*. Bd. 1, IEEE Computer Society, 12 2009. – ISBN 978–0–7695–3888–4, S. 414 –417 (Zitiert auf den Seiten 24 und 43)

[LS05] Lahl, Uwe ; Steven, Wilhelm: Feinstaub - eine gesundheitspolitische Herausforderung. In: *Pneumologie* 59 (2005), S. 704–714. http://dx.doi.org/10.1055/s-2005-915558. – DOI 10.1055/s–2005–915558. – 0934-8387 (Zitiert auf Seite 15)

[Mac92] MacEachren, Alan M.: Visualizing uncertain information. In: *Cartographic Perspective* (1992), Nr. 13, S. 10–19 (Zitiert auf Seite 20)

[Mur07] Murugesan, San: Understanding Web 2.0. In: *IT Professional* 9 (2007), july-aug., Nr. 4, S. 34 –41. http://dx.doi.org/10.1109/MITP.2007.78. – DOI 10.1109/MITP.2007.78. – ISSN 1520–9202 (Zitiert auf Seite 12)

[OM02] Olston, Chris ; Mackinlay, Jock D.: Visualizing Data with Bounded Uncertainty. In: *Proceedings of the IEEE Symposium on Information Visualization*. Washington, DC, USA : IEEE Computer Society, 2002 (INFOVIS '02). – ISBN 0–7695–1751–X, 37–40 (Zitiert auf den Seiten 17, 26 und 31)

[Pra09] Prasanna, Dhanji R.: *Dependency Injection*. 1st. Greenwich, CT, USA : Manning Publications Co., 2009. – ISBN 193398855X, 9781933988559 (Zitiert auf Seite 13)

[Qui03] Quint, Antoine: Scalable vector graphics. In: *Multimedia, IEEE* 10 (2003), july-sept., Nr. 3, S. 99 – 102. http://dx.doi.org/10.1109/MMUL.2003.1218261. – DOI 10.1109/MMUL.2003.1218261. – ISSN 1070–986X (Zitiert auf Seite 33)

[Riv07] Riveiro, Maria: Evaluation of uncertainty visualization techniques for information fusion. In: *10th International Conference on Information Fusion*, 2007, 1-8 (Zitiert auf den Seiten 15, 16, 17, 20, 22 und 30)

[THM+05] Thomson, Judi ; Hetzler, Elizabeth ; MacEachren, Alan ; Gahegan, Mark ; Pavel, Misha: A typology for visualizing uncertainty. In: Erbacher, R. F. (Hrsg.) ; Roberts, J. C. (Hrsg.) ; Gröhn, M. T. (Hrsg.) ; Börner, K. (Hrsg.): *Visualization and Data Analysis 2005. Edited by Erbacher, Robert F.; Roberts, Jonathan C.; Gröhn, Matti T.; Börner, Katy. Proceedings of the SPIE, Volume 5669, pp. 146-157 (2005).* Bd. 5669, 2005 (Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series), 146–157 (Zitiert auf den Seiten 16, 17 und 27)

[TWHS04] Theisel, Holger ; Weinkauf, Tino ; Hege, Hans-Christian ; Seidel, Hans-Peter: Stream Line and Path Line Oriented Topology for 2D Time-Dependent Vector Fields. In: *Proceedings of the conference on Visualization '04*. Washington, DC, USA : IEEE Computer Society, 2004 (VIS '04). – ISBN 0–7803–8788–0, 321–328 (Zitiert auf Seite 12)

[Van08] Vanbrabrant, Robbie: *Google Guice: Agile Lightweight Dependency Injection Framework*. Apress, 2008 (FirstPress Series). – ISBN 9781590599976 (Zitiert auf Seite 13)

[WSF+95] Wittenbrink, Craig M. ; Saxon, Elijah ; Furman, Jeff J. ; Pang, Alex ; Lodha, Suresh: Glyphs for visualizing uncertainty in environmental vector fields. In: *IEEE Transactions on Visualization and Computer Graphics* Bd. 2410, 1995, 266–279 (Zitiert auf Seite 20)

Alle URLs wurden zuletzt am 18.08.2011 geprüft.

Erklärung

Hiermit versichere ich, diese Arbeit selbständig verfasst und nur die angegebenen Quellen benutzt zu haben.

(Stefan Wokusch)